The MagPi

Issue 107 | July 2021 | magpi.cc

The official Raspberry Pi magazine

BECOME A **RASPBERRY PI**

GENIUS

Troubleshooting tips for everyone

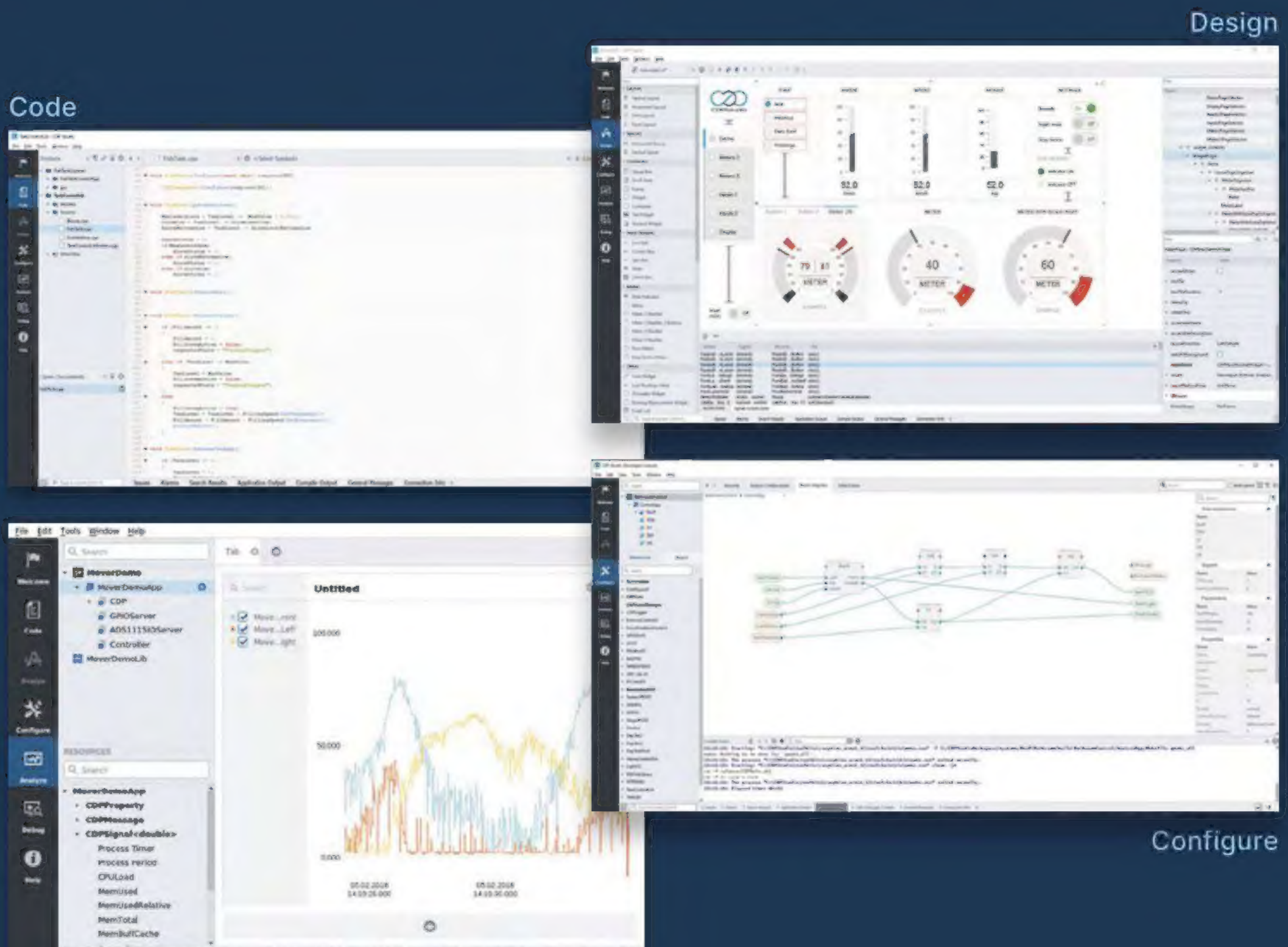
Starter problems Solved!

Make a **Weather Watcher**

Recreating The Mars Helicopter

Brilliant Handheld projects

50 PAGES OF PROJECTS & TUTORIALS



PROFESSIONAL CONTROL SYSTEM DEVELOPMENT TOOL

Home projects made easy.

CDP Studio, a great software development tool for your home projects. Build systems for Raspberry Pi, use C++ or NoCode programming, open source libraries, out of the box support for GPIO, I2C, MQTT, OPC UA and more. Create beautiful user interfaces. Built for industrial control system development, **FREE for home projects**.

cdpstudio.com
Tel: +47 990 80 900 • info@cdptech.com
CDP Technologies AS // Hundsværgata 8, 6008 Ålesund, Norway

CDPStudio

WELCOME
to *The MagPi* 107

Raspberry Pi is an amazing computer that enables you to explore computing from a whole different perspective. It enables you to check out the technology, right down to the metal.

Become a Raspberry Pi Genius (page 32) is this month's lead feature, and is all about fixing the problems that trip up newcomers. Nobody likes to think of themselves as a 'genius', but understanding how a computer works is a form of magic all in itself.

Rob has got more Big Builds this month. Only now they're Handheld Builds (page 72). These projects are big in scope, small in stature. Tiny arcade machines, laptop computers, and digital cameras. They are amazing projects that don't need a massive garage to build.

Speaking of arcades, there was a collective intake of breath when K.G submitted the photos of their arcade build with the decals affixed (page 42). The result looks nothing short of incredible.

There are a lot of things that 'wow' in this edition of *The MagPi*. I hope you enjoy it as much as us.

Lucy Hattersley Editor

EDITOR

Lucy Hattersley

Lucy is editor of *The MagPi* and magically transforms coffee into code.

@LucyHattersley

Contents

Issue 107 July 2021

Cover Feature

32 Become a Raspberry Pi Genius

Regulars

92 Your Letters
97 Next Month
98 The Final Word

Project Showcases

10 Epigone Drone
14 Temperature monitor
18 Apple iPad Magic Mirror
20 Bluebot Shoal Fish Robot
22 MiniSplitPi
24 Pico Pong
26 RUHAcam
28 Plant Rotator

Bluebot Shoal Fish Robot

RUHAcam

Tutorials

42 Build an arcade machine - part 4
48 Make a Weather Watcher
52 Pico-Voice - part 2
56 Create GUIs in Python - part 5
62 Configure Pi-hole - part 3
66 Pico temperature gauge

The Big Feature

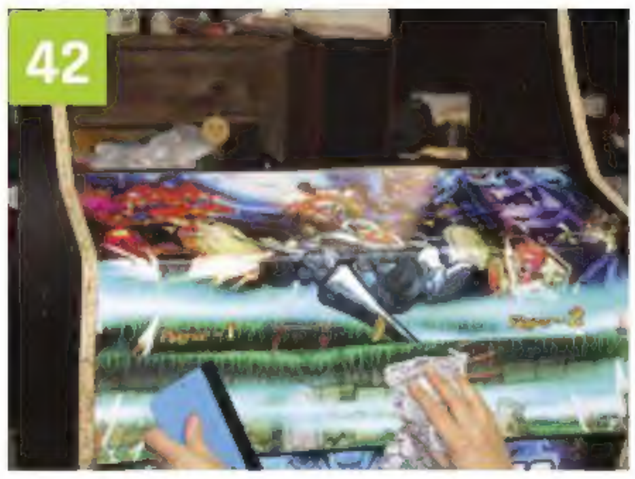
Handheld builds

Reviews

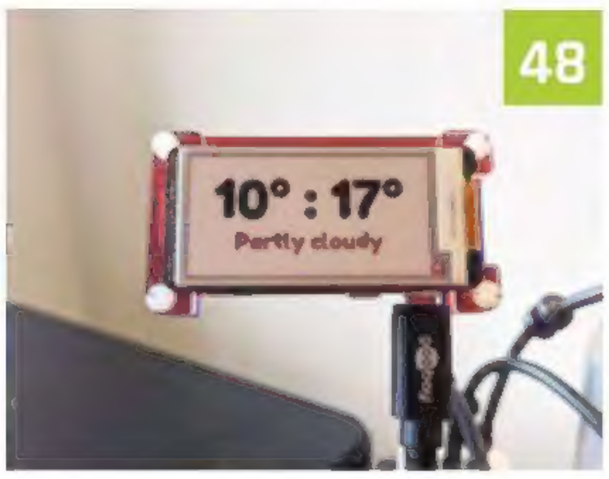
78 Marty The Robot v2
80 micro:Pi
82 10 Amazing: Displays
84 Learn Linux

Community

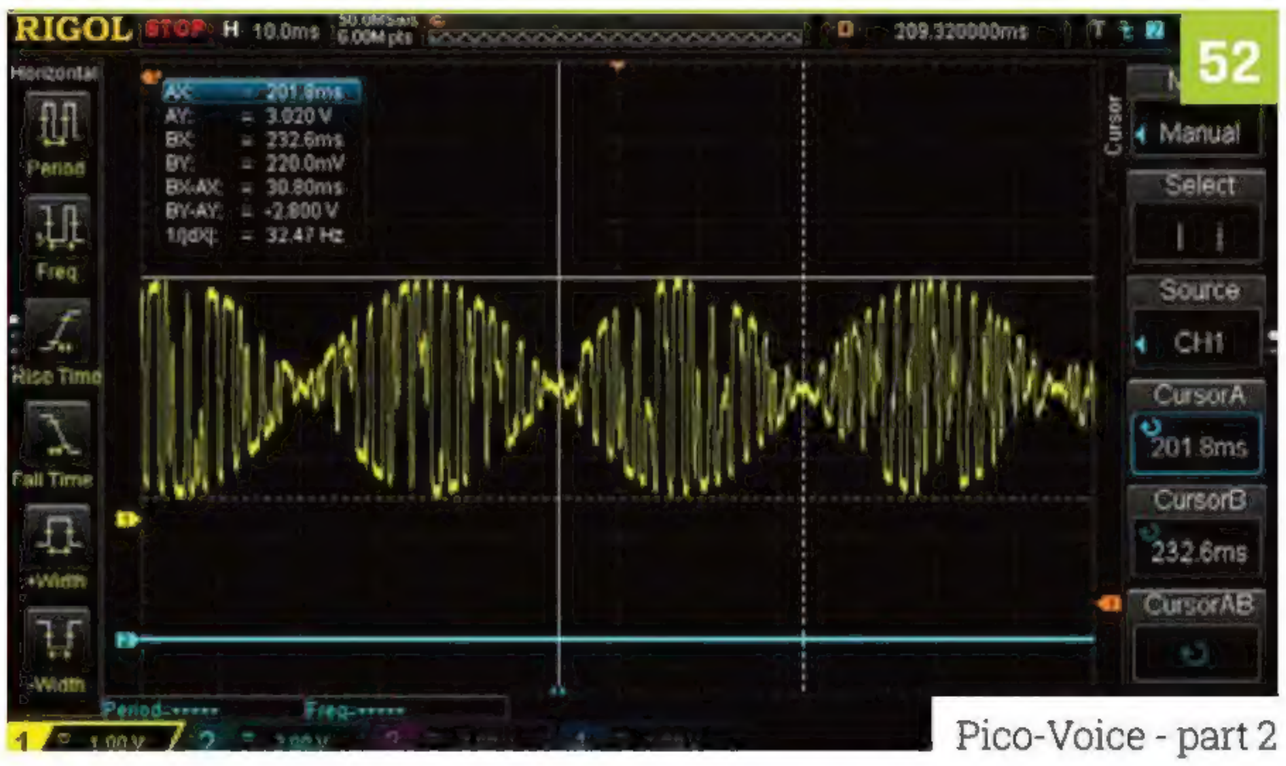
86 Al Pemartin interview
88 This Month in Raspberry Pi



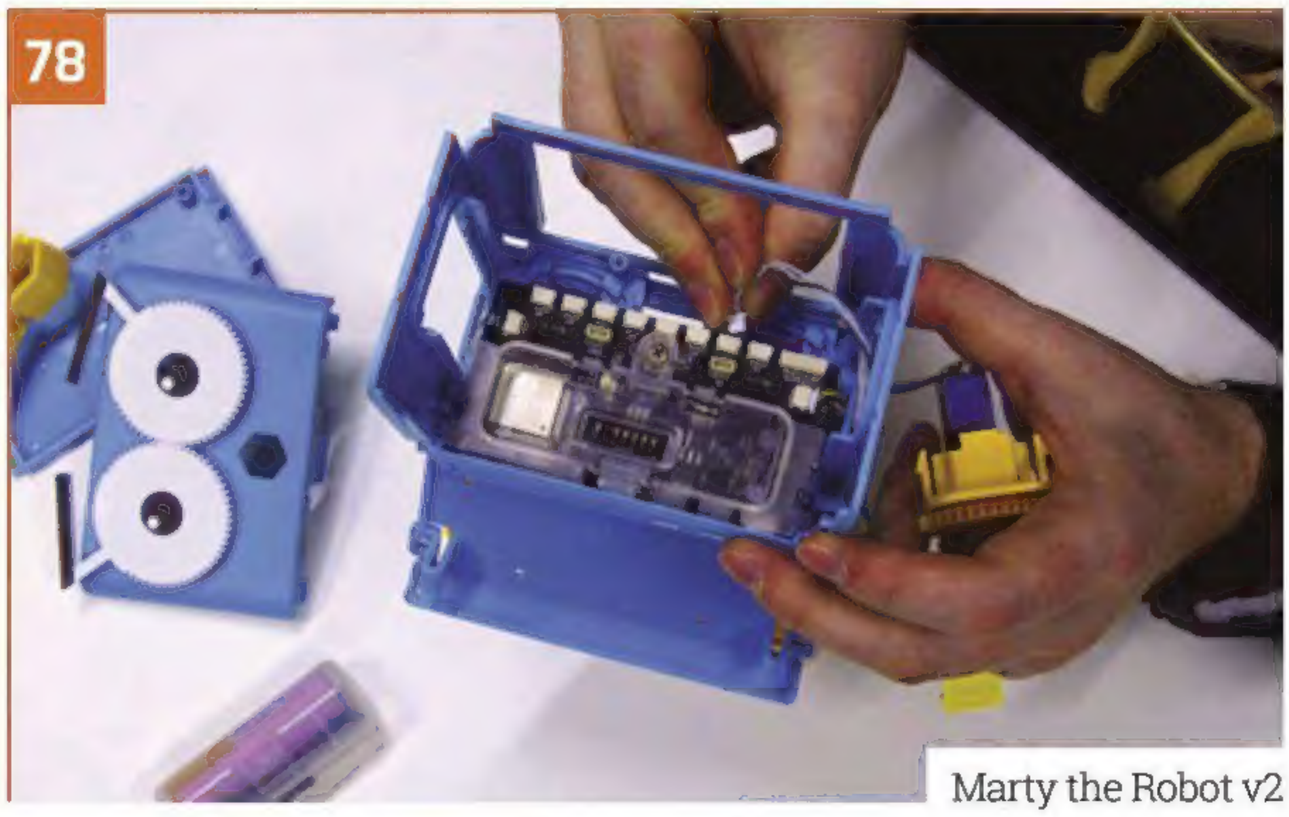
Build an arcade machine - part 4



Make a Weather Watcher



Pico-Voice - part 2



Marty the Robot v2

WIN 1 OF 5 THINE CAMERA **EXTENSION KIT** 95

Power armour

for your Raspberry Pi 4

Mini OLED screen & programmable buttons

Integrated 1-watt loudspeaker

Integrated heat sink & centrifugal cooling fan

5-hour internal battery

UPS-style soft shutdown system prevents SD card corruption

USB-C Power Delivery specification at 15V for faster battery charging

Full access to Raspberry Pi's GPIO pins & ports

The ultimate Raspberry Pi case to go anywhere and do anything!

pi-top [4] DIY Edition is the only Raspberry Pi 4 case with an inbuilt battery and power management system. It docks to a family of 'plates' to connect to our expanding range of sensors, motors and more.

By installing your Raspberry Pi 4 into **pi-top [4] DIY Edition**, it is upgraded into the ultimate mobile hands-on computing platform. Build anything from rovers to drones: the only limit is your imagination!

pi-top [4] DIY Edition - **from £94.95 / $99.95**

Raspberry Pi is a trademark of the Raspberry Pi Foundation.
*Not photoshopped. We really did make a pi-top drone!

pi-top Raspberry Pi made simple, robust and modular. pi-top.com/MagPi

Raspberry Pi PoE+ HAT announced

Power your Raspberry Pi 4 straight from the Ethernet connection

▲ Raspberry Pi PoE+ HAT connects directly to Raspberry Pi via the GPIO pins

Raspberry Pi has announced the next generation of its Power over Ethernet (PoE) HAT.

Power over Ethernet is a technology that enables you to power a device from an Ethernet connection (providing the Ethernet network has power-sourcing technology).

Raspberry Pi PoE+ HAT (**magpi.cc/poe+**) delivers more power, implementing the 802.3at PoE+ standard; and it runs cooler, thanks to various design improvements. "Best of all, we've been able to keep the original affordable price of $20 [£18 in the UK]," says Eben Upton, Raspberry Pi creator and founder of the Raspberry Pi Foundation.

More power

The original PoE HAT could deliver a guaranteed minimum of 13W to a Raspberry Pi. Enough to power Raspberry Pi 4 at maximum load, but not quite enough to also power the hungriest USB peripherals.

> Raspberry Pi PoE+ HAT delivers more power... and it runs cooler

The PoE+ HAT implements the newer 802.3at standard. When used with a compatible switch or injector, it can deliver up to 25W.

"To reduce heat dissipation, we replaced the diode rectifier with an 'ideal diode' rectifier, in the form of a Microchip PD70224ILQ device," explains Eben.

"For the PoE+ HAT, our friends at Bourns have provided us with a shiny new toy: a planar transformer." You can learn more about these technologies in Eben's blog post on Raspberry Pi's website (**magpi.cc/announcingpoe+**).

Raspberry Pi PoE+ HAT is available now from SB Components, OKdo, and other Raspberry Pi resellers (**magpi.cc/poe+**).

RP2040 on sale now at $1

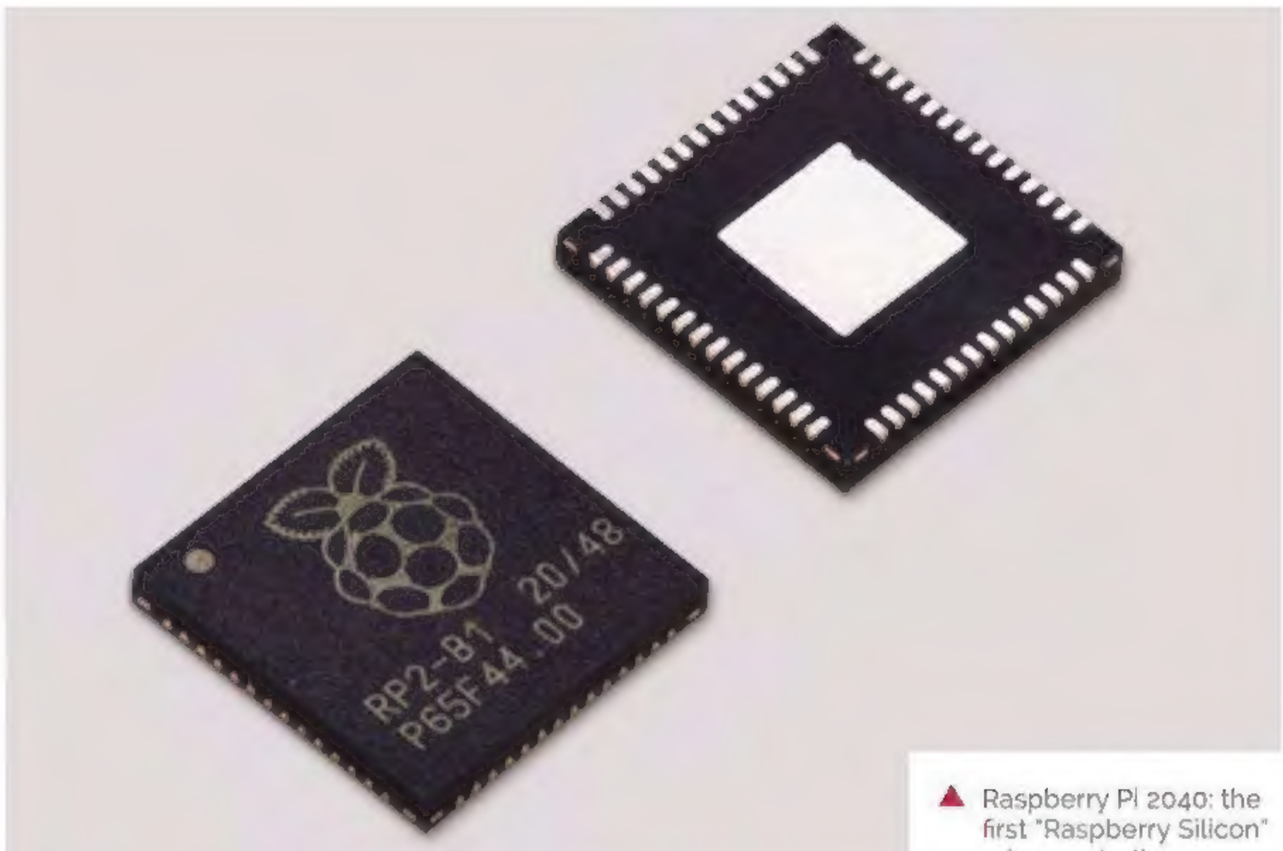
Buy the heart of Raspberry Pi Pico for a buck

RP2040 chips are now available from Approved Resellers in single-unit quantities, allowing you to build your projects and products on Raspberry Silicon.

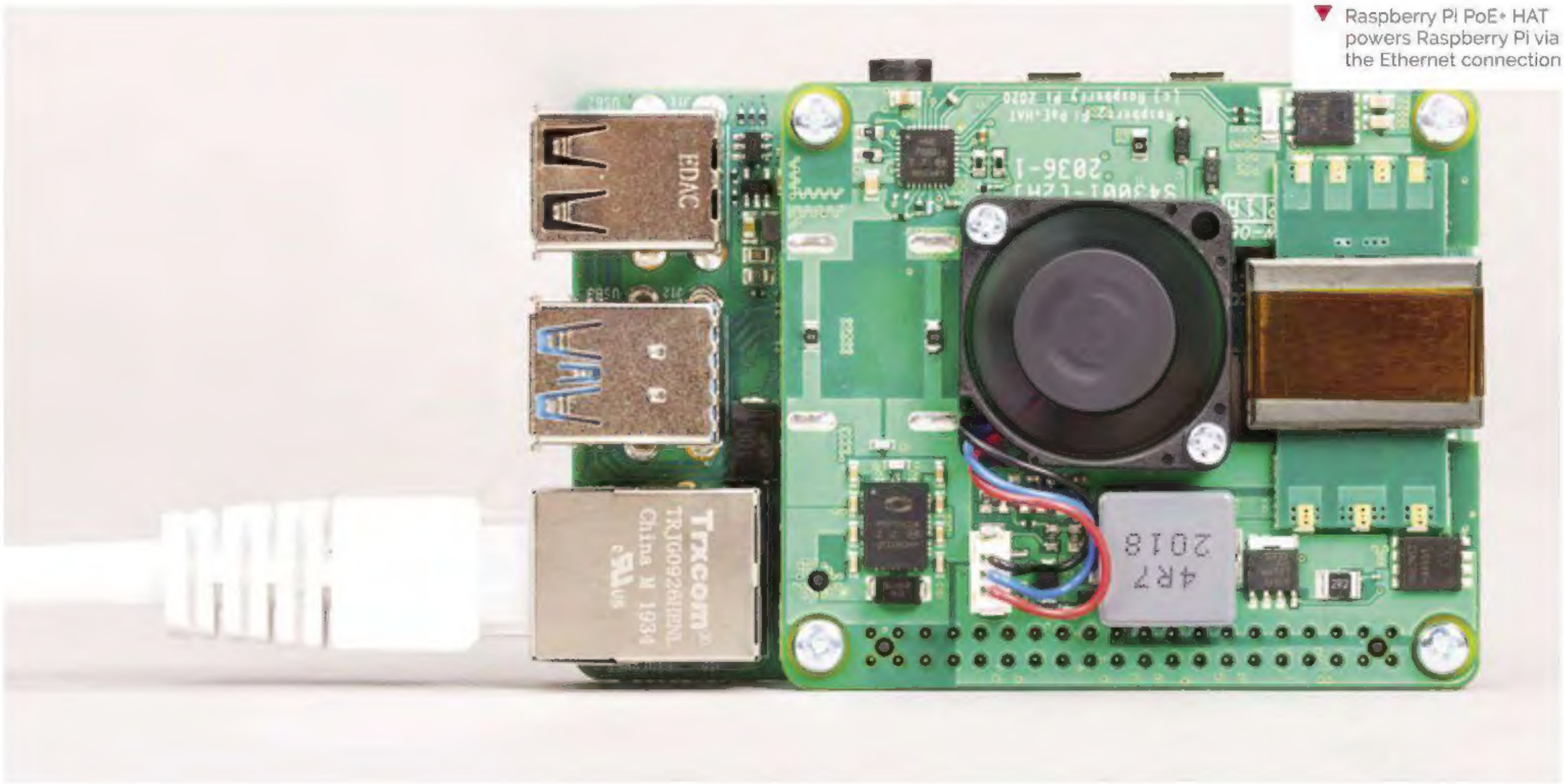
Raspberry Pi Pico was the board that introduced RP2040 to the world (magpi.cc/pico). "RP2040 is our idea of the perfect mid-range microcontroller, based on years of using other vendors' devices in our own products and projects," says Eben Upton.

"The single-unit price of RP2040 is $1, giving you a lot of bang for your (literal) buck. We're still figuring out what reel-scale pricing [bulk purchasing] will look like in the autumn, but we expect it to be significantly lower than that."

Head on over to the R2040 product page (magpi.cc/rp2040) to order your first chips. When you're ready to take your RP2040-based project to scale, Raspberry Pi will be waiting for you.



▲ Raspberry Pi 2040: the first "Raspberry Silicon" microcontroller



▼ Raspberry Pi PoE+ HAT powers Raspberry Pi via the Ethernet connection

Epigone Drone

Inspired by NASA's attempt to launch a helicopter on Mars, one maker made an Earth-bound one of her own, hears **Rosie Hattersley**

MAKER

Avra Saslow

Avra studied computer science and geography in Boulder, Colorado, having got hooked on electronics and Raspberry Pi while making glass speakers for her final physics project.

instagram.com/avranator47

Like millions of us, in April Avra Saslow watched with bated breath as NASA's Perseverance rover touched down on the surface of Mars.

Like most of us, Avra knew all about the other ground-breaking feat being trialled alongside Perseverance: a helicopter launch called Ingenuity, that was to be the first flight on another planet – "a fairly lofty goal", says Avra, since "the atmosphere on Mars is 60 times less dense than Earth's."

With experience of Raspberry Pi-based creations, Avra was keen to emulate Ingenuity back here on earth (**magpi.cc/ingenuity**).

NASA chose to use open-source products and use commercially available parts for its helicopter build. It just so happened that Avra had recently begun working at SparkFun, a Colorado-based reseller that sells the very same Garmin LIDAR-Lite v3 laser altimeter that NASA's helicopter is based on. "It's a compact optical distance measurement sensor that gives the helicopter 'eyes' to see how far it hovers above ground," Avra explains.

A Garmin LIDAR-Lite v4 LED altimeter acts as 'eyes' and measures changes in the surface elevation

To avoid being swiped by the drone's rotors, the Raspberry Pi 4, which uses NASA's especially written F Prime code for telemetry, had to be positioned very carefully

The Epigone piggybacks on a borrowed DJI drone and is able to fly successfully for several minutes, despite a load equal to its own weight

FACTS

- Avra is a big fan of Python for citizen science projects
- As a keen cyclist, Epigone's mapping plays to her love of wild terrains
- Epigone means 'a less distinguished follower or imitator'
- Avra loves Perseverance/Ingenuity for its diversity of makers
- Ingenuity's lead engineer helped Avra complete her own version

▲ NASA recognises that Raspberry Pi offers a way to "dip your toe in embedded systems," says Avra, and "encourages the idea that Linux can run on two planets in the solar system"

NASA posted the Ingenuity helicopter's open-source autonomous space-flight software on GitHub. It was written specifically for use with Raspberry Pi!

▲ NASA's Ingenuity helicopter, which inspired the Epigone, completes its first one-way flight on Mars. (Image credit: NASA)

Alert!
Drone safety

Be mindful of spinning blades when modifying drones. Drone usage in the UK is regulated by the Civil Aviation Authority. Make sure you read the The Drone and Model Aircraft Code before flying your drone (and research relevant drone regulations in other countries)

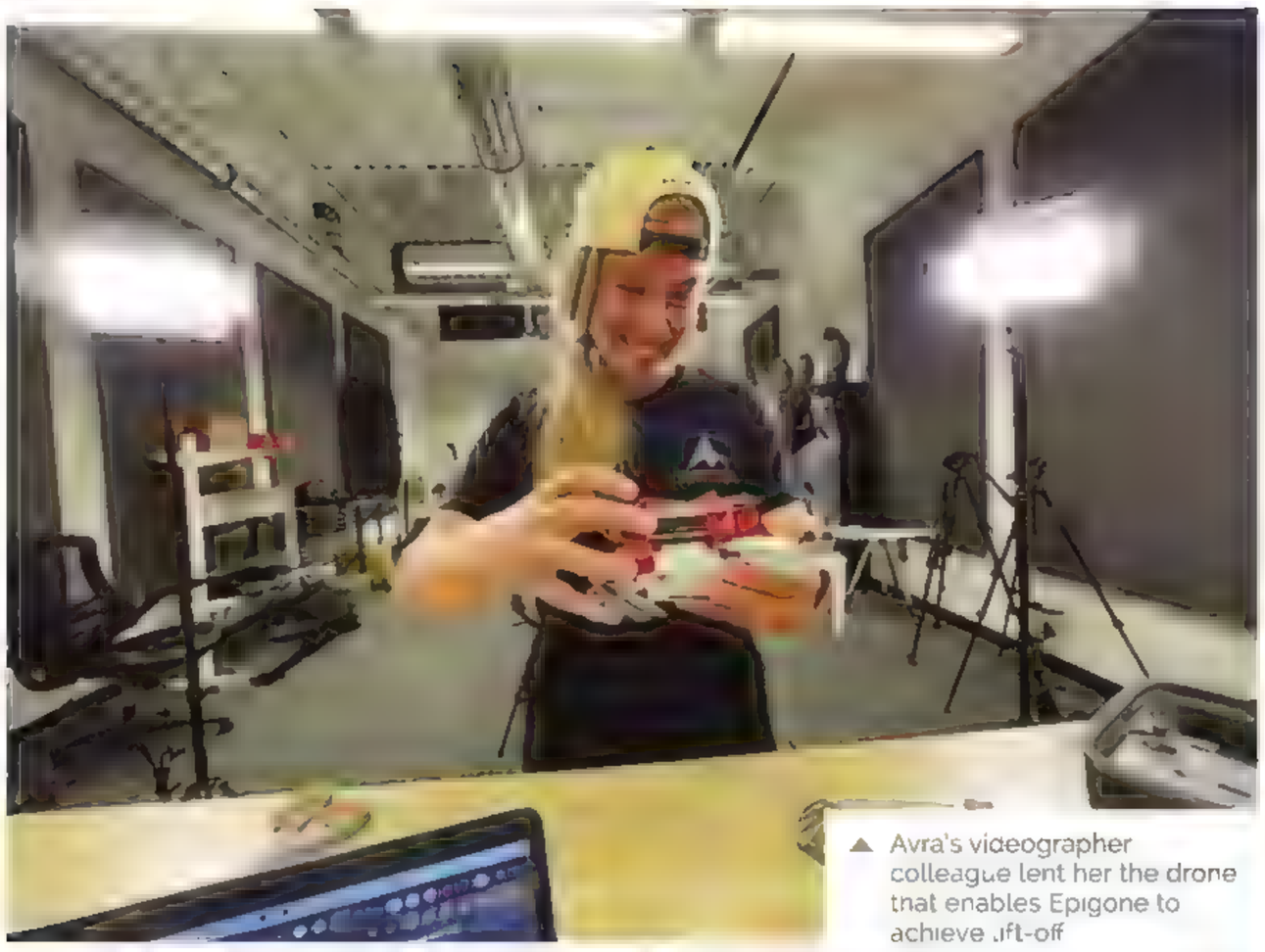
magpi.cc/dronecode

NASA posted the Ingenuity helicopter's open-source autonomous space-flight software, written specifically for use with Raspberry Pi, on GitHub. Avra took all this as a sign she "just had to experiment with the same technology they sent to Mars."

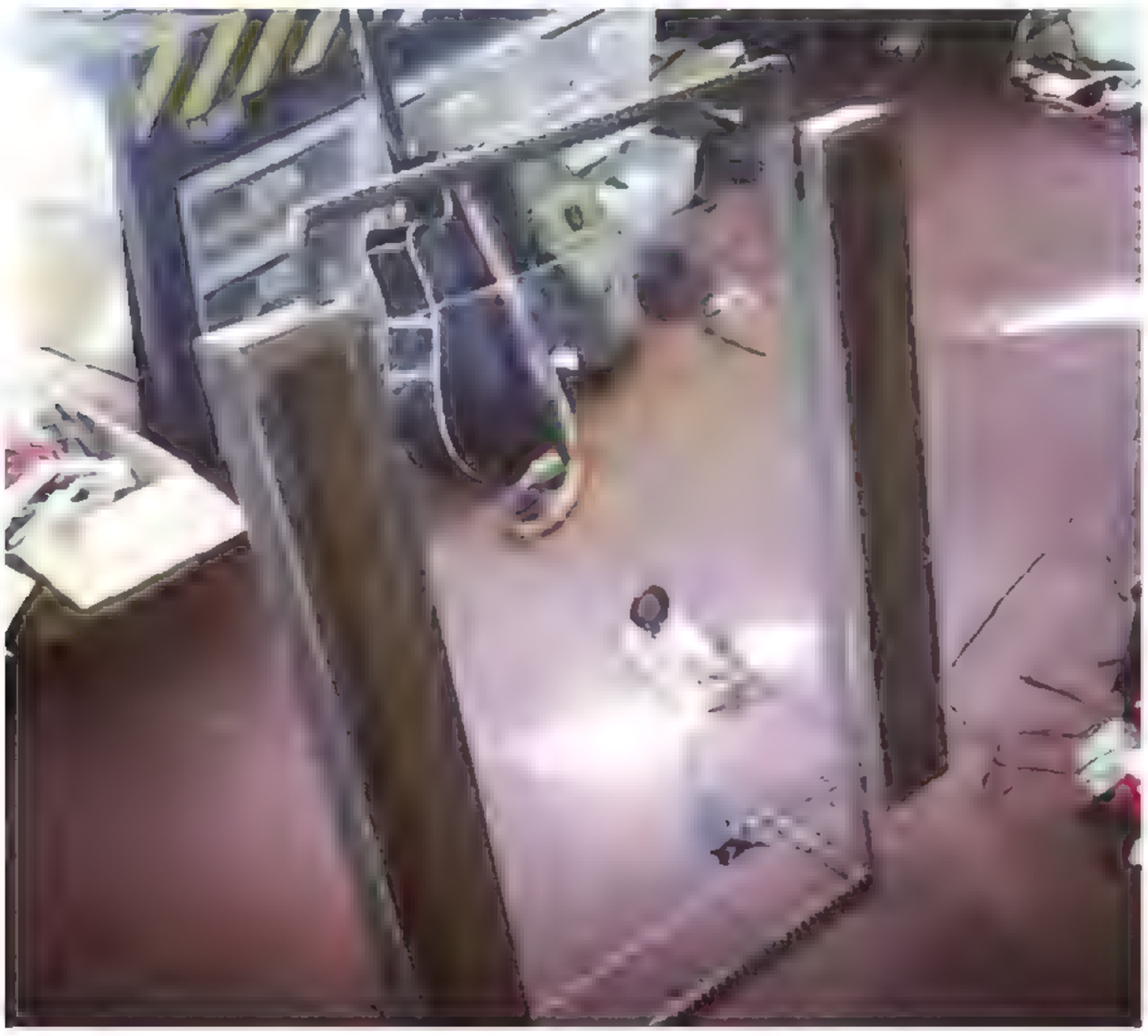
F Prime and shine

Her plan was to see whether she could get GPS and lidar working within NASA's framework, "and then take the sensors up on a drone and see how it all performed in the air." Helpfully, NASA's GitHub post included a detailed F Prime tutorial based around Raspberry Pi: **magpi.cc/fprimegit**. Avra says understanding and using F Prime (F´) was the hardest part of her Epigone drone project. "It's a beast to take on from an electronics enthusiast standpoint," she says. Even so, she emphatically encourages others to explore F´ and the opportunity to make use of NASA's code: **magpi.cc/fprime**.

The Epigone Drone is built around Raspberry Pi 4 Model B; Garmin's LIDAR-Lite v4, which connects to a Qwiic breakout board and has a laser rather than an LED; a battery pack; and a DJI Mini 2 drone borrowed from a videographer colleague. Having seen how small the drone was, Avra realised 3D-printing an enclosure case would make everything far too heavy. As it was, positioning the Epigone onto its host drone was challenging enough: the drone's rotors passed worryingly close to the project's Raspberry Pi, even when precisely positioned in the centre of the drone's back. The drone has its own sensors to allow for controlled navigation, which meant Avra's design had to diverge from NASA's and have its lidar 'eyes' on its side rather than underneath.



▲ Avra's videographer colleague lent her the drone that enables Epigone to achieve lift-off



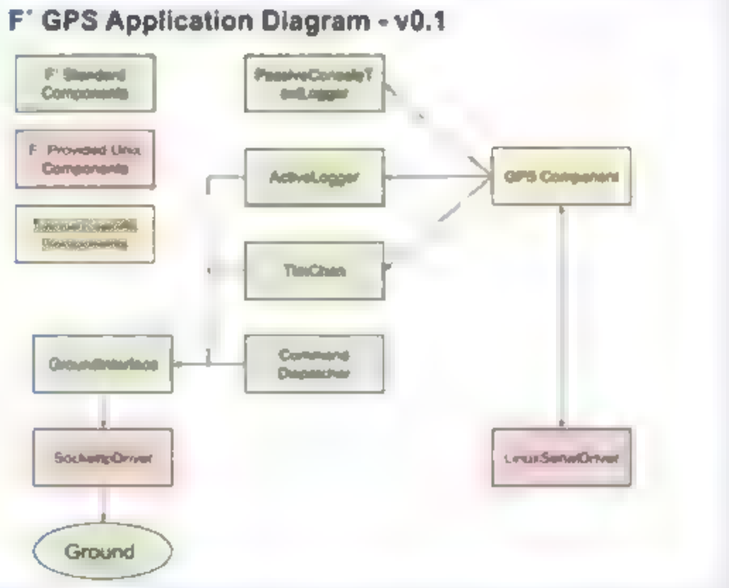
Although her version piggybacks on an existing drone, Avra was amazed when her Epigone creation took flight. "I honestly thought [it] would be too heavy to achieve lift, but what do ya know, it flew! It went up maybe 30 ft and we were able to check the sensors by moving it close and far from the SparkFun HQ [where she works]."

While the drone's battery depleted in "a matter of minutes" due to its additional load, the Epigone worked well and could be deployed to map small areas of land such as elevation changes in a garden, Avra suggests.

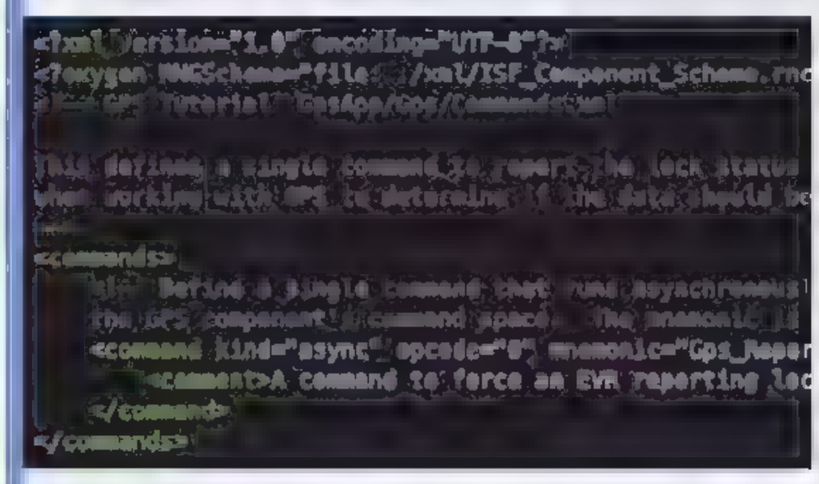
▲ These glass speakers, made for her physics final project, were Avra's first foray into Raspberry Pi and Arduino

Make a Mars helicopter

01 To emulate Avra's version, you need Raspberry Pi 4, a Garmin LIDAR-Lite v4 for telemetry, an NMEA GPS receiver, Qwiic breakout board, a power pack, and a drone to provide the lift. You should closely follow NASA's detailed F' tutorial for Raspberry Pi (**magpi.cc/fprimegit**). The steps assume you have your own drone to power your craft.



02 NASA explains how to use F' with Raspberry Pi and an NMEA-enabled GPS receiver attached to a serial driver (**magpi.cc/fprimegps**). It can read in GPS messages from a UART port, then produce events and telemetry through the GPS link.



03 Avra built a lidar component using NASA's GPS template to create the component through XML, then deployed the component on her native host and cross-compiled it for Raspberry Pi.

Temperature monitor

A simple, cheap, and contactless solution for taking people's temperatures with Raspberry Pi. **Rob Zwetsloot** checks it out

Philip Koreski

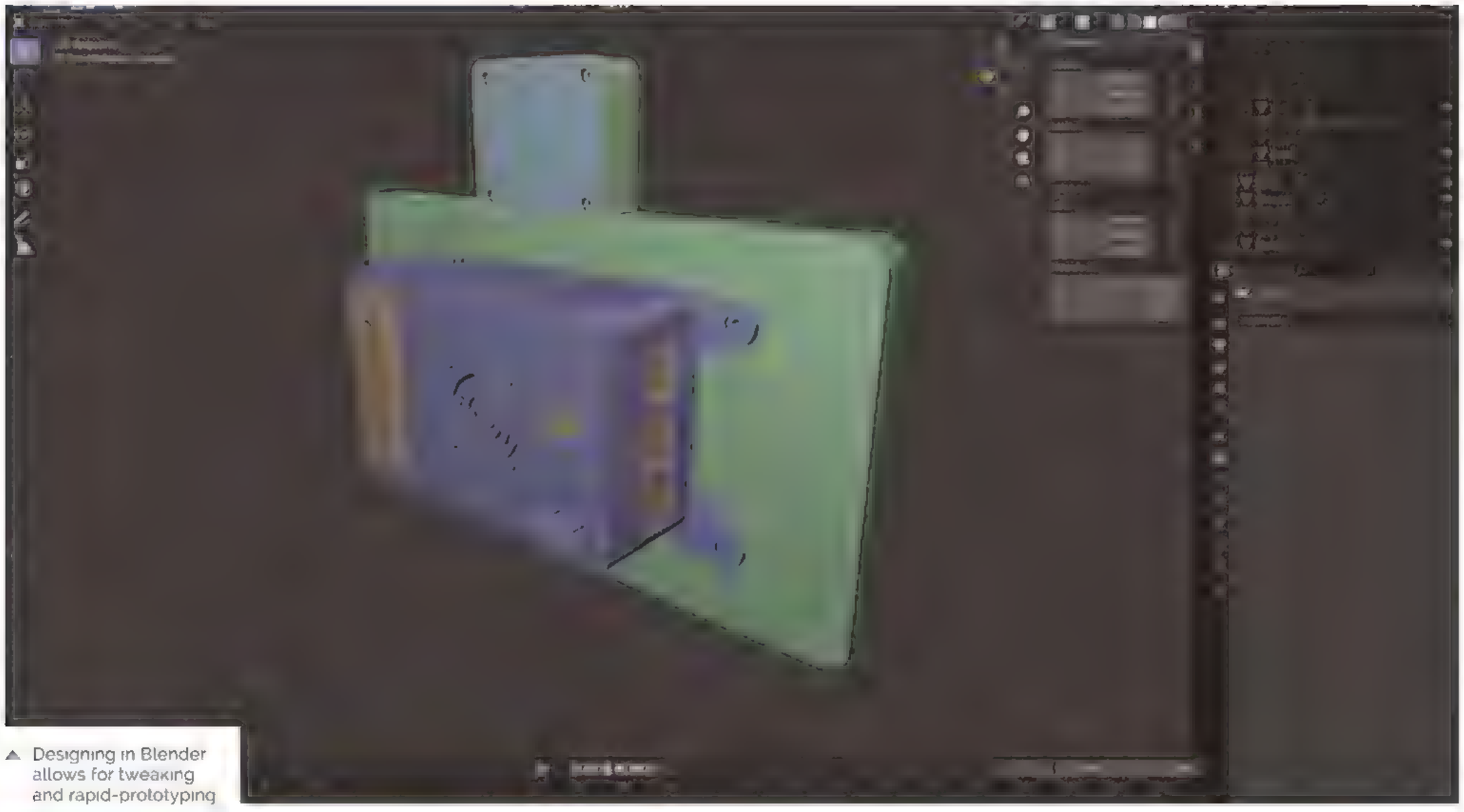
A programmer in the healthcare informatics department at Pacific Cataract and Laser Institute in Chehalis, Washington. He primarily customizes electronic healthcare records software and develop applications to improve clinical workflow efficiency.

Over the past 18 months, a lot of workplaces have introduced temperature checks for employees and customers, and healthcare fields are no different. While some use expensive hand scanners, Philip Koreski figured out a different solution: "I recently designed a touch-free temperature reading station for determining whether the employees in our office are fit for work or not, based on their temperature."

At Philip's workplace, a laser eye surgery, employees would need to get their temperature taken and then fill out a paper form to confirm that they could work.

"I was given the assignment to digitise this process using a touchscreen device," he tells us. "I thought that it would be nice to combine the process of taking our temperature with filling out the form. The other thing is that everyone who used it would have to touch the screen, so I figured if facial recognition was incorporated, it would be more hygienic. So what was developed is a Raspberry Pi-based device with an IR temperature sensor that can recognise the employee

- 37°C/98.6°F is only an average for adult body temperature.
- It usually ranges between 36.1°C/97°F and 37.2°C/ 99°F
- The monitor uses a contactless MLX90614 IR sensor
- The case was 3D-printed and designed in Blender
- A Haar cascade is a type of object recognition used in face detectors



▲ Designing in Blender allows for tweaking and rapid-prototyping

> The facial recognition works well to identify the registered users and the temperature reader is pretty accurate too

▲ The 3D-printed case is quite simple, and easy to mount on a wall

▶ A Raspberry Pi speaker, and display connector are squeezed into the case

in front of it, take their temperature, and log whether it is too high or acceptable for work."

Cool design

The touchscreen integrated into the monitor also allows for manual entry if required.

"Raspberry Pi is an obvious choice for a project like this," Philip explains. "It is compatible with a wide range of sensors, has the processing power to be able to handle things like facial recognition, and it has the conveniences that come with a full-blown operating system. On top of that, it has great support for Python and most of the libraries needed straight out of the box. I do a lot of my software development in Python, so this is a major plus."

Development happened over the course of three-to-four weeks as a side project, programming, part sourcing, and 3D design for the case.

"The largest part of that was actually hand gesture recognition believe it or not," Philip says. "I wanted a good way to signal to the device that it got the identity of the user wrong. It took a lot of trial and error, but eventually I found a Haar cascade for palm recognition and that seemed like a good way to tell it to 'stop'. But the nice thing is that something relatively complex, like facial

▲ The monitor is mounted to a wall so you have to walk past it

recognition and temperature reading and logging, can be done by one developer in a reasonably short amount of time."

Hot product

With projects like this, there's always a concern that it won't do its job. That was not the case for this monitor.

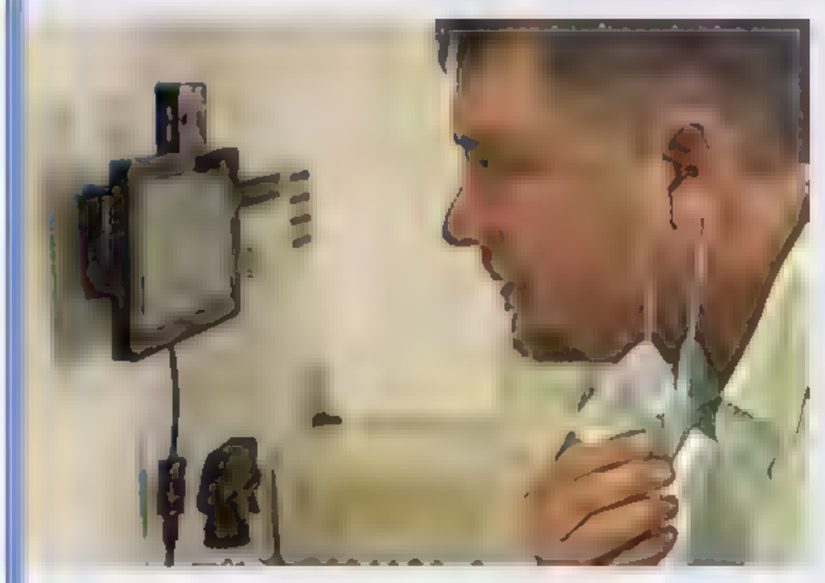
"The facial recognition works well to identify the registered users and the temperature reader is pretty accurate too," Philip reveals. "In testing, we've found that distance from the sensor plays a large roll in accuracy, so we will be working on ways for the users to have an easier time placing themselves in the correct range of distance from the sensor. Overall, the response has been pretty good from the users."

Currently, the monitor is still being tested, but, if it passes, Philip hopes to get it rolled out throughout other departments in the company, and hopefully further.

▼ A manual mode can be used to override the automated system if required

Temperature check

01 First, the device needs to recognise the user. This happens in under a second, however it gives a five-second window which allows for manual change if there's a mis-identification.



02 The IR sensor requires the user to be a lot closer than the camera, so the user steps up to the machine and places their head a couple of inches away so an accurate temperature can be taken and recorded.

03 The system analyses the data and gives the user a pass or fail. If the system isn't working properly, a manual mode can be enabled where a user selects themself and then confirms whether or not they're fit to work.

Apple iPad Magic Mirror

Salvaging a smashed first-gen iPad by combining it with Raspberry Pi Zero. **David Crookes** reflects on it

▲ The screen displays lots of information from a website set up by Chris, who is considering creating a smaller mirror

Chris Greening

Chris Greening is a maker based in Edinburgh. He really enjoys building projects with embedded electronics and sharing what he's learnt with the community

magpi.cc/ipadmirror

Apple launched the iPad in 2010, but the first-generation tablet was discontinued the following year and there were no more iOS updates available for it after 2012. So, many of these debut devices were either binned, sold on, or consigned to drawers. Yet, as Chris Greening proves, Raspberry Pi can blow new life into them.

"My iPad was so old that apps didn't really work any more and the Safari browser wasn't performing well with modern websites," he says. "The glass was also cracked and the batteries didn't charge, so it was heading to the bin." Or at least it was until Chris decided to use it as the basis for a magic mirror.

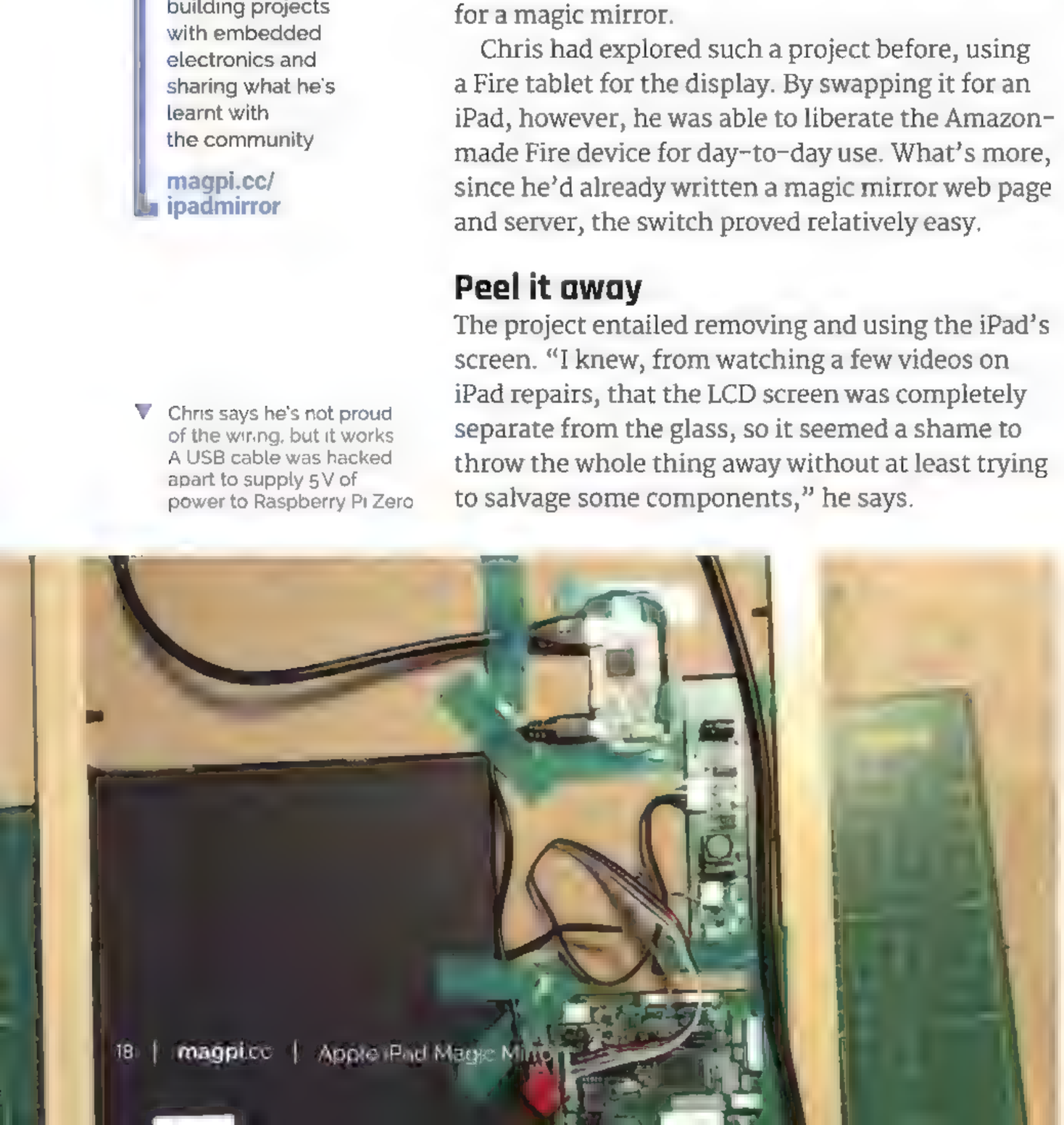
Chris had explored such a project before, using a Fire tablet for the display. By swapping it for an iPad, however, he was able to liberate the Amazon-made Fire device for day-to-day use. What's more, since he'd already written a magic mirror web page and server, the switch proved relatively easy.

Peel it away

The project entailed removing and using the iPad's screen. "I knew, from watching a few videos on iPad repairs, that the LCD screen was completely separate from the glass, so it seemed a shame to throw the whole thing away without at least trying to salvage some components," he says.

Unfortunately, this was the most difficult part of the build. "It's a lot easier if the glass is not already cracked," he explains. "In theory, you should just be able to pop the glass out from the metal frame, and using a thin metal spudger seems to be the way to go. As my glass was already cracked, I went with a much more brute force approach."

Once he'd isolated the screen, it was then a case of hooking it up to a Raspberry Pi Zero W. This required a driver board. "The main thing to check is that it's definitely compatible with the first iPad's LCD screen which has the model number LP097X02," Chris says. "The driver boards to seem to be fairly standard now."



▼ Chris says he's not proud of the wiring, but it works A USB cable was hacked apart to supply 5V of power to Raspberry Pi Zero

The core business

Raspberry Pi Zero W – powered using a 5V buck converter and running Raspberry Pi OS – proved to be a perfect match. Being thin, it fits nicely behind the mirror. "It's also so affordable but can still run a full operating system with a GUI and it has wireless LAN, too – those are probably the most important things for this project because I really needed to be able to run a web browser," Chris says.

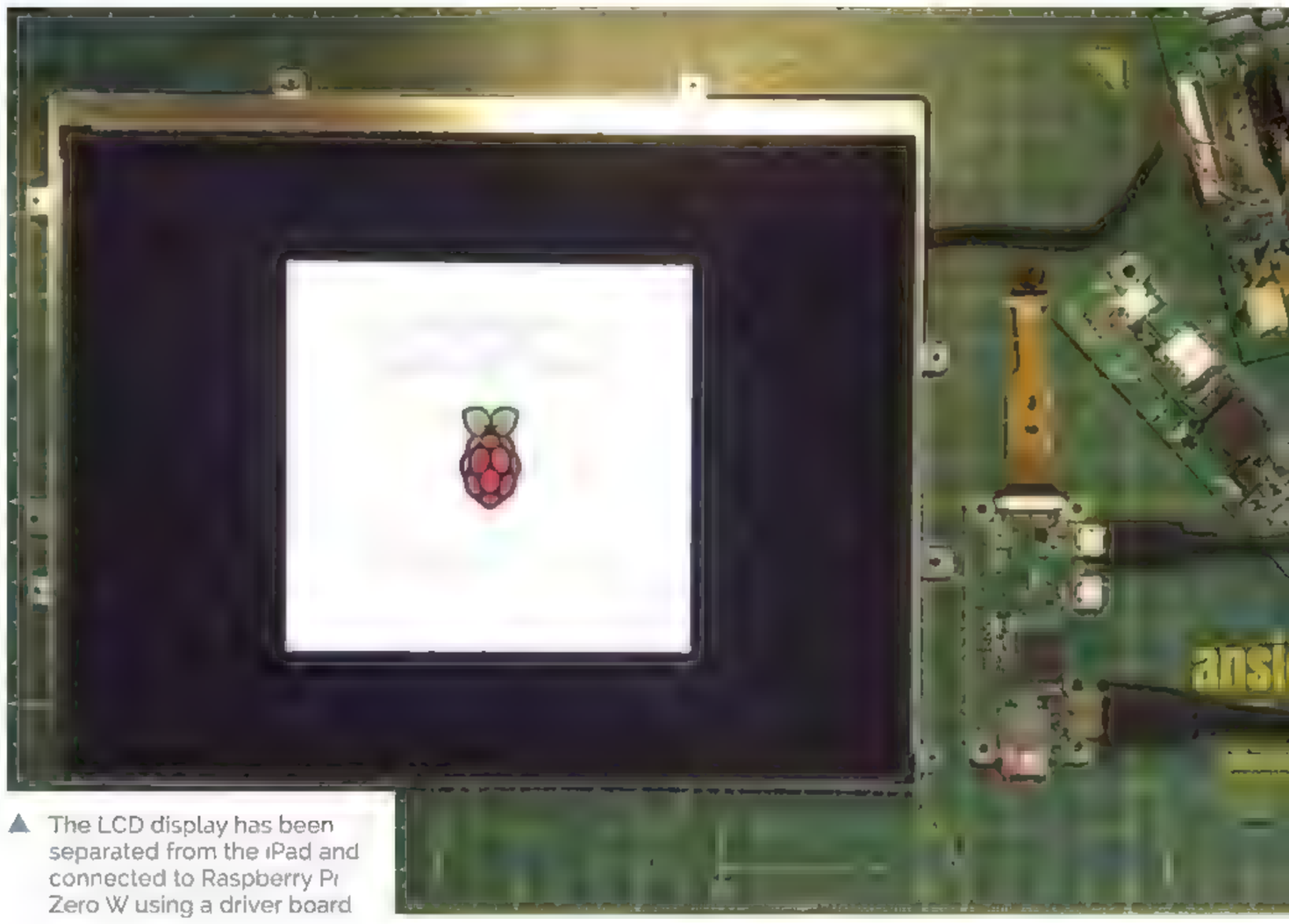
After setting the Chromium browser to auto-start in full-screen kiosk mode, it was pointed at Chris's magic mirror URL. "But there is some really

> There is some really nice magic mirror software available for Raspberry Pi

nice magic mirror software available for Raspberry Pi, so I may look at switching over at some point," he muses. From that point, it was a matter of fitting everything together.

"The main challenge was ensuring the components wouldn't stick out from behind the mirror too much," he reveals. "The driver boards come with VGA and composite connectors which are quick thick compared to the actual PCBs." There was also a problem with the screen's brightness, solved using tinted acetate sheets, but otherwise everything went to plan.

"You can pick up refurbished, second-hand iPad LCDs on eBay for very good prices, especially if they have a couple of dead pixels," Chris says. "You can create a very affordable display."



▲ The LCD display has been separated from the iPad and connected to Raspberry Pi Zero W using a driver board

Bluebot Shoal Fish Robot

If you loved the film Finding Dory, you might just enjoy the original story of these underwater robots, suggests **Rosie Hattersley**

Florian Berlinger

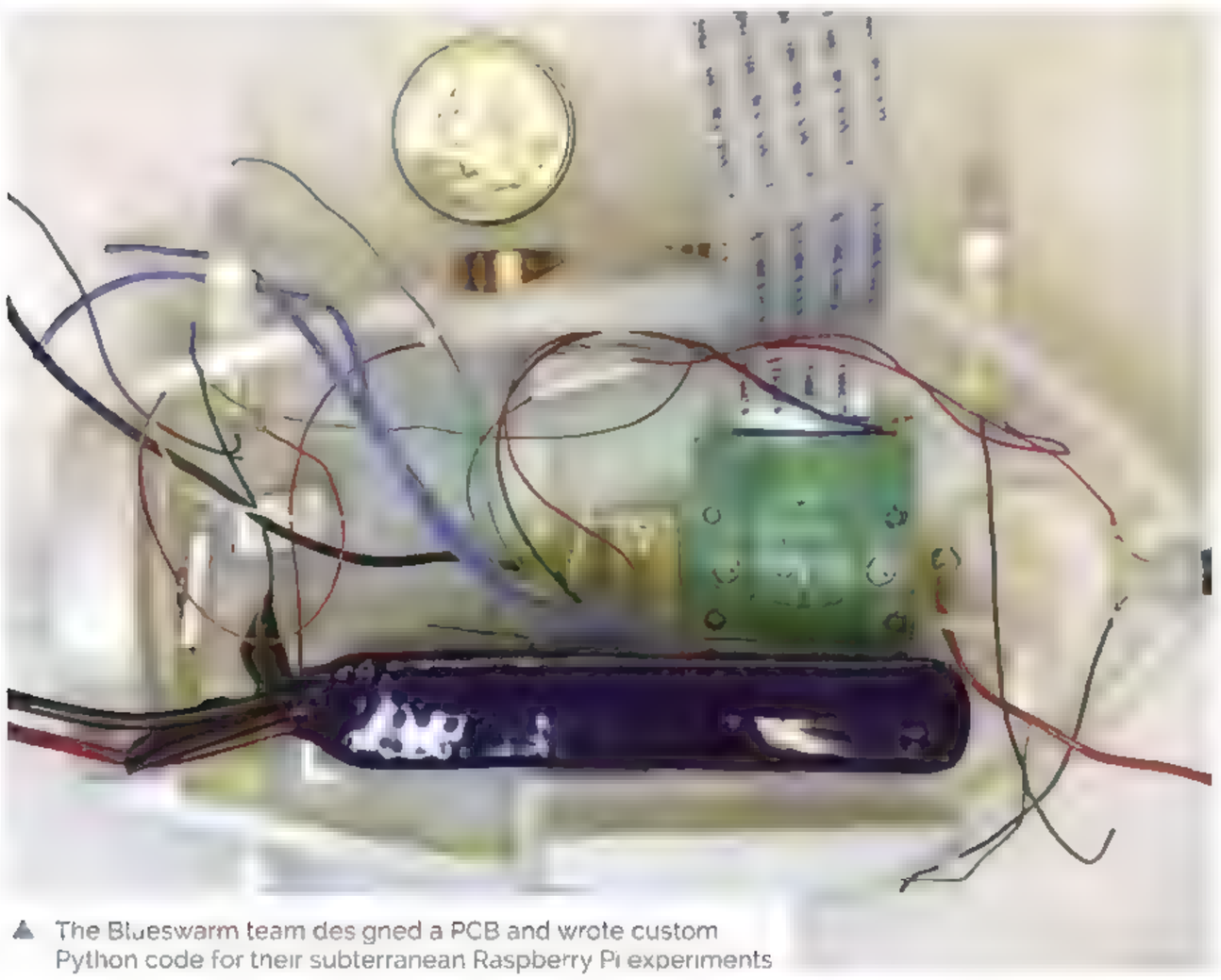
Florian is fascinated by displays of collective intelligence and co-operation such as schools of fish and teams of robots.

florianberlinger.ch

It's no coincidence that the shoal of robot fish in this Raspberry Pi Zero W project look more than a little like Dory from Pixar's movie.

As with the film character, the Bluebot robot fish are based on the blue tang or surgeonfish. Unlike Dory, however, these robot fish are designed to be anything but loners. They behave collectively, which is the focus of the Blueswarm research project that began in 2016 at Harvard University.

Florian Berlinger and his PhD research project colleagues Radhika Nagpal, Melvin Gauci, Jeff Dusek, and Paula Wulko set out to investigate the behaviour of a synchronised group of underwater robots and how groups of such robot fish are co-ordinated by observing each other's movements. In the wild, birds, fish, and some animals co-ordinate in this way when migrating, looking for food and as a means of detecting and collectively avoiding predators. Simulations of such swarm behaviour exist, but Blueswarm has the additional challenge of operating underwater. Raspberry Pi Zero W works well here because multiple Bluebot robots can be accessed remotely over a secure wireless connection, and Raspberry Pi Zero W is physically small and light enough to fit inside a palm-sized robot.



▲ The Blueswarm team des gned a PCB and wrote custom Python code for their subterranean Raspberry Pi experiments

Mimicking movements

The team designed the fish-inspired, 3D-printed robot body as well as the fin-like actuators and the on-board printed circuit board which connects to all the electronics and communicates with Raspberry Pi Zero W. Designing the robot fish took the team four years, from working out how each robot fish would move and adding sensing capabilities, to refining the design and implementing collective behaviours, coded using Python 3.

They used as many off-the-shelf electronics as possible to keep the robots simple, but adapted existing software algorithms for the purposes of their investigations, "with several clever twists on existing algorithms to make them run fast on Raspberry Pi," adds Florian.

On-board cameras that offer "an amazing 360-degree field of view" are one of the project's real triumphs. These cameras are connected to Raspberry Pi via a duplexer board (so two cameras can operate as one) the project team co-designed with Arducam (see **arducam.com**). Each Raspberry Pi Zero W inside follows the camera images and instructs the fins to move accordingly. The team developed custom algorithms for synchronisation, flocking, milling, and search behaviours to simulate how real fish move individually and as a group. As a result, says Florian, "Blueswarm

- There's a video about the Bluebot shoal on YouTube (**magpi.cc/swarmoffish**)
- The Bluebots are already being used for other studies on marine behaviour
- Blueswarm are going to study open water marine behaviour next
- When tackling complex projects, break them into smaller tasks.
- Start simple, develop and test each stage individually, Florian advises

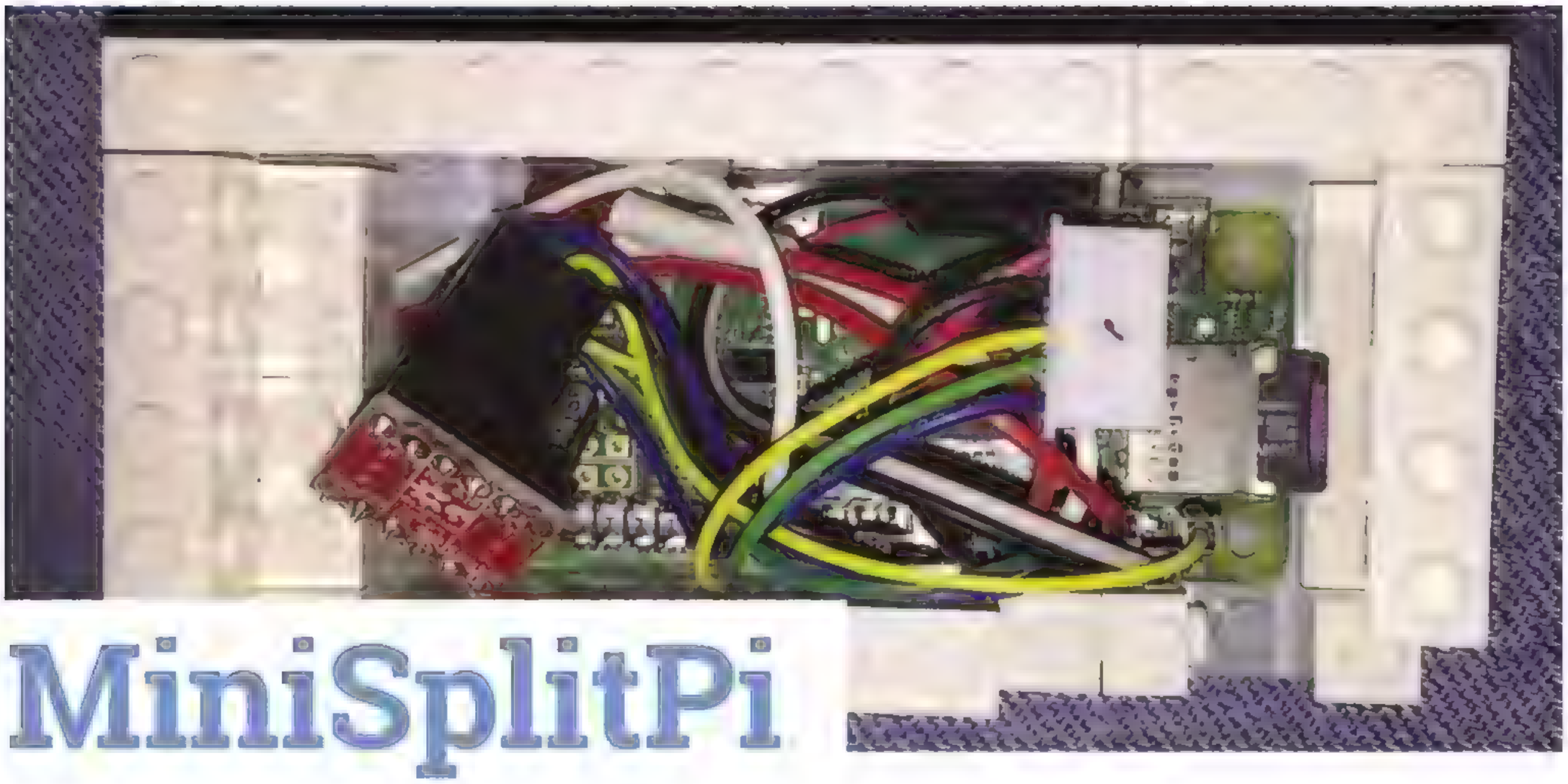
> On-board cameras offer an amazing 360-degree field of view

can be used to study inter-robot co-ordination in the laboratory and to learn more about collective intelligence in nature." He suggests other robot-based projects could make use of a similar setup.

Imitation of life

Each robot fish cost around $250 and took approximately six hours to make. To make your own, you'd need a 3D printer, Raspberry Pi Zero W, a soldering station – and a suitably large tank for your robot shoal! Although the team hasn't made the code available, the Blueswarm project paper has recently been published in Science Robotics and by the IEEE Robots and Automation Society (**ieee-ras.org**). Several biology researchers have also been using the Bluebot shoal as 'fish surrogates' in their studies of swimming and schooling.

▲ It may look cute, but Bluebot has a serious purpose



MiniSplitPi

Controlling a commercial mini-split heat pump using a Raspberry Pi. **Rob Zwetsloot** cranks up the heat on this cheaper solution

▲ It's a fairly simple Raspberry Pi Zero setup inside the case

Robert H'obbes' Zakon

An entrepreneur, technology consultant, and a former White House Presidential Innovation Fellow. He has a passion for using tech to make lives better

magpi.cc/minisplitpi

"After installing a photovoltaic (solar) system with battery backup, managing home loads during grid outages became important," Robert H'obbes' Zakon says in his write-up for his MiniSplitPi. It's a remote-control device powered by Raspberry Pi that controls a mini-split heat pump, which is a kind of heating and cooling unit used in some homes.

"Mini-split heat pumps use a lot of energy when it is really cold out, and are the one home device we have that may continue to run unnoticed during an outage, such as when we are asleep or out of the house, causing the batteries to drain down rapidly," he explains further. "So, finding a way to automatically shut them off at the start of an outage became the goal of this project, and in the process ended up with full web control of the mini-splits. Yes, they could have been put on a separate panel that is not powered during an outage. However, what's the fun in that? We still wanted the ability to run them for A/C during the summer when they don't use up much energy."

Cheaper alternative

While there are commercial products that perform this function, they tend to cost around $200 (£140), whereas Robert's Raspberry Pi version costs only $20 (£14) for each device. They also wouldn't perform the shut-off during the grid outage that he originally planned to control.

The MiniSplitPi is connected via a CM105 port on the heat pumps, which are usually used with the official remote control products.

"The CN105 connector pins provide 12V, ground, 5V, TX, and RX," Robert explains. "Communications are through the serial UART (TX/RX). Although the 5V could possibly be used to power Raspberry Pi, in an abundance of caution the 12V was used instead, along with a buck converter to step down the voltage to the required 5V for Raspberry Pi. Similarly, the CN105 TX/RX are 5V, whereas Raspberry Pi requires 3V, so a bidirectional logic level converter was used."

Robert added other features, such as a shutdown button if wireless contact is lost, an LED for at-a-glance checks of the grid, and a motion-activated screen with more data and adjustable settings.

Hot stuff

"The MiniSplitPi has worked remarkably well," Robert says. "Being able to control all of the mini-splits from a single web interface has made life easier. The one gotcha I have found, and am still chasing down, is trying to switch multiple heads from heating to cooling mode at the same time. This is usually only required twice a year, so turning them off then on again in the new mode, something that can be automated, is not a big issue. The primary reason for developing the MiniSplitPi has worked extremely well, though. As

This mini-split heat pump does have remote control capabilities; however, they don't do everything Robert wanted

Raspberry Pi is connected to a port inside the mini-split's control system so it can control it

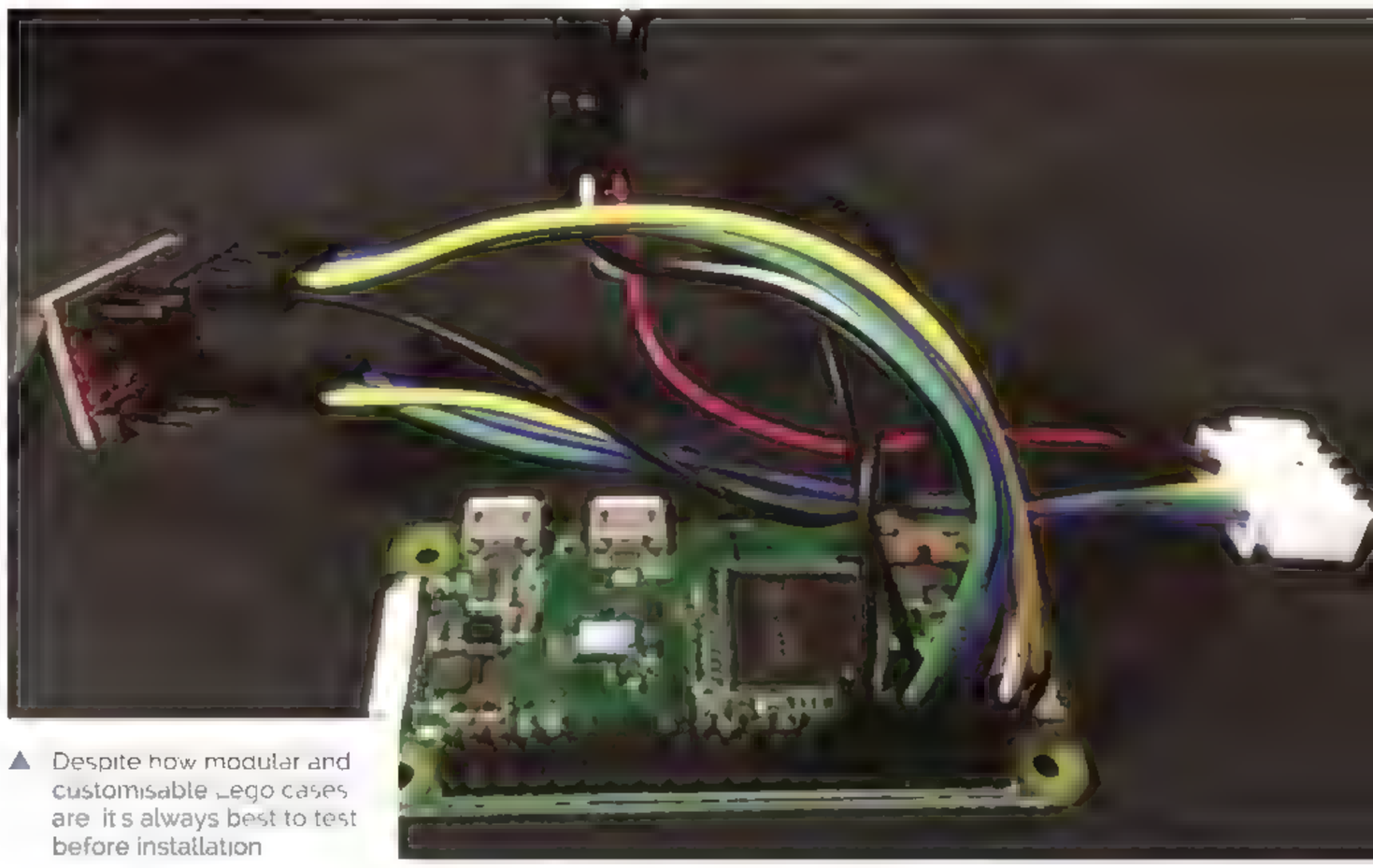
Wireless LAN allows the whole system to be operated from a browser on the network

- Robert previously made the AI-powered Seeing Wand
- Raspberry Pi Zero W is one of his favourite micro computers
- The outputs were decoded to create a Python library...
- ... which you can find here **magpi.cc/mqmitsi**
- The full build includes a custom API if you plan to replicate it

> Being able to control all of the mini-splits from a single web interface has made life easier

our home is now powered by photovoltaics with Tesla Powerwall batteries for backup, I wanted a way to have the mini-splits go off automatically after ten minutes if the grid goes down, so the batteries are not quickly drained. The MiniSplitPi worked as expected for the power outages we had over this past winter, and it has been useful to have the LED on it showing an active outage, as it is not always evident when the grid goes down."

Plans for a big touchscreen to work as a thermostat are in the works as well, so that it's more accessible to family members.



▲ Despite how modular and customisable Lego cases are it's always best to test before installation

Pico Pong

Recreating an arcade classic with gesture control on Raspberry Pi Pico. **Nicola King** connects bat with ball

Nick Bild

A software engineer by day, Nick is a hardware hacker by night. His formal education is in computer science and bioinformatics: he is self-taught in electronics

magpi.cc/picopong

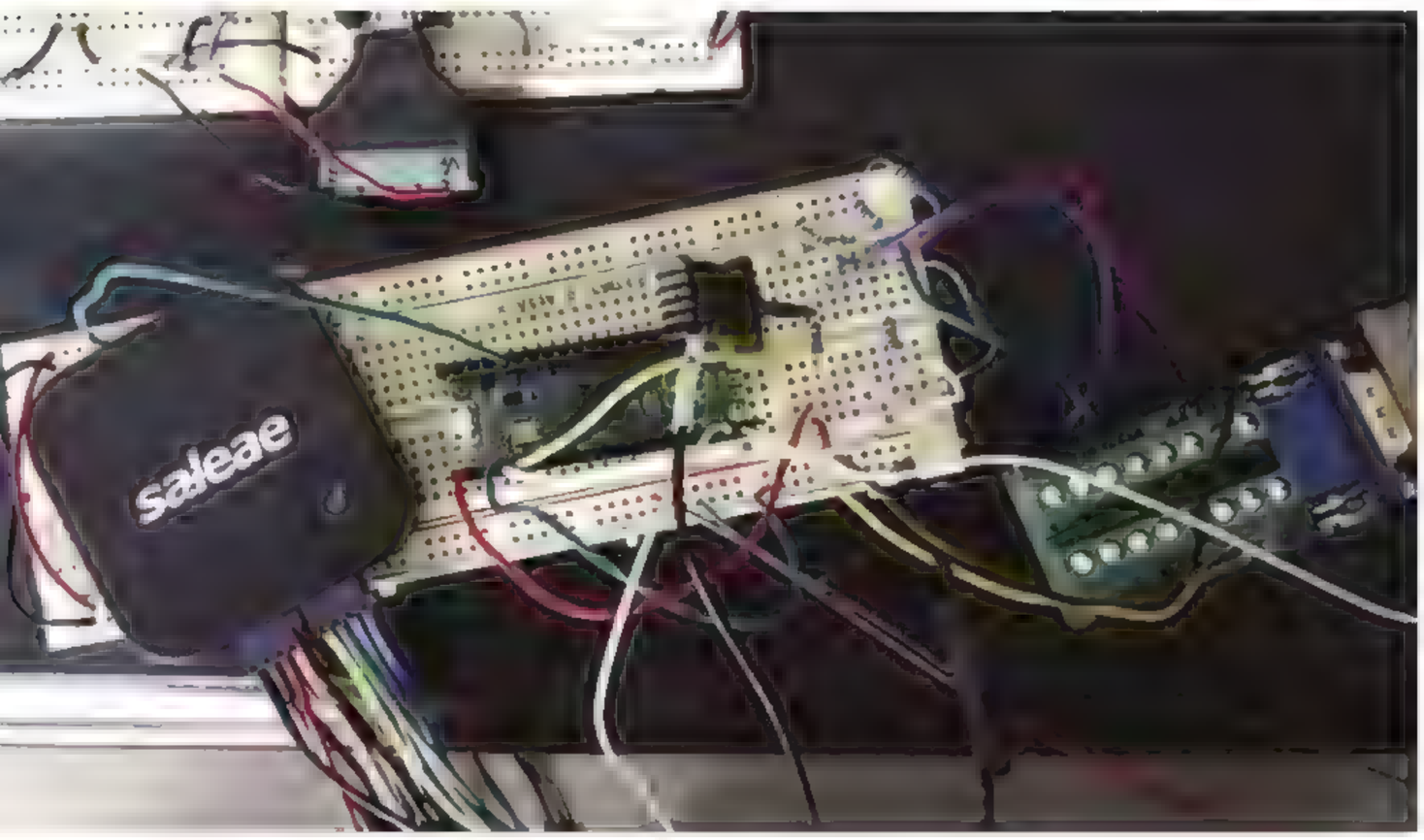
After discovering that Raspberry Pi Pico can use its Programmable Input/Output (PIO) capability to output a video signal, Nick Bild was inspired to recreate the arcade classic Pong. Not only that, but he's made it gesture-controlled.

"I love playing old games in RetroPie on my Raspberry Pi 4 with my son Luke," he tells us, "so when I realised what a perfect fit Pico's PIO would be for generating a VGA signal, my mind went to a reimagining of one of the greatest games of all time."

It took Nick about three weeks of nights and weekends to build Pico Pong: "I have a day job, so I have to fit my projects in when I can."

The difficult part was getting the VGA signal timing just right. "Once I had that locked down, it only took me a couple of hours to code up the Pong algorithm and graphics," he reveals.

Since Pico's PIO handles the cycle-intensive process of generating video, this leaves the CPU free to do other work, like play Pico Pong. "I use the PIO to generate a 640×480@60Hz monochrome VGA signal," says Nick. "Data for the display is stored in a buffer in SRAM, which the PIO program grabs at exactly the right nanosecond, as it paints the screen, via DMA [direct memory access]."

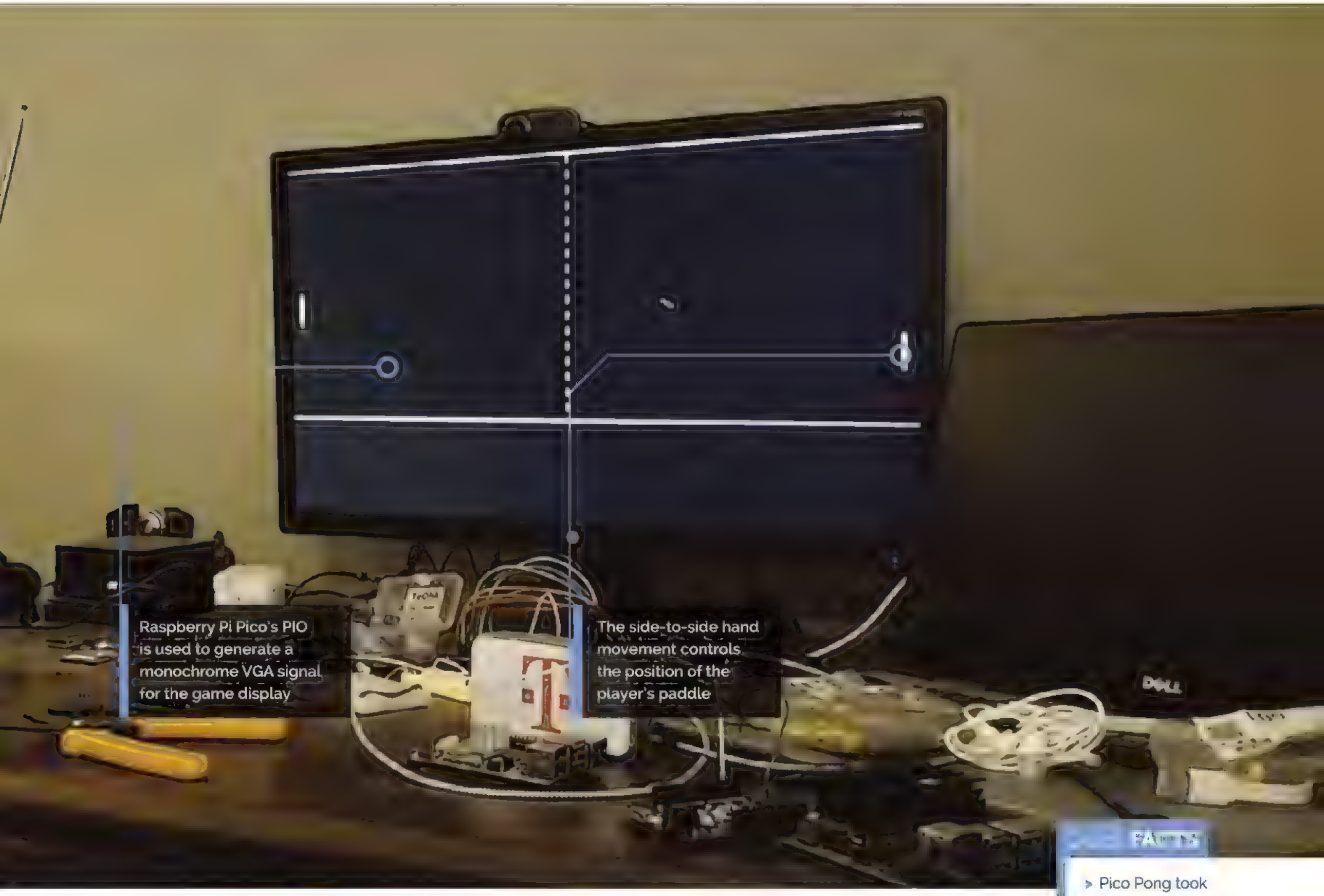


▼ Much of the project development involved debugging the VGA signal to get it just right

As well as overclocking Pico to 258MHz, nearly double its standard speed, it took a lot of incremental tweaking to get the VGA signals exactly right. "I had to reorganise and rethink the PIO assembly code a number of times to get the data for each pixel on the GPIOs exactly within its 40 nanosecond window."

Gesture control

The player's on-screen paddle is gesture-controlled using two infrared (IR) LED / IR phototransistor pairs. "The IR LED shines upward onto a mirror, which reflects the IR light back onto the receiver below," explains Nick. "When that signal is interrupted (i.e. by a hand), it triggers Pico via GPIO to move the player paddle. One receiver



moves the paddle up on the screen, and the other moves the paddle down."

Why gesture control? "I wanted to do something different from the typical potentiometer controller, so I did some digging through my parts bins to get ideas," Nick replies. "I first thought I might use a PIR sensor for a similar gesture interface, but even with hardware modifications it wasn't responsive enough."

As an alternative, he tried the IR LED/phototransistor pair and responses were smooth and instantaneous. "After playing around with this for a while, I feel like this is the way Pong was meant to be played. If my idea weren't over 40 years late, I think this could have been the Nintendo Wii of the 1970s."

Game, set, and match

After showcasing the project on Hackster.io (**magpi.cc/picopong**), Nick has received plenty of positive feedback from the maker community.

> I've found that many people are drawn to this new way of playing a classic game

"There's a lot of love for Pong, so I've found that many people are drawn to this new way of playing a classic game," he says. "People see this as something that they could realistically build for themselves, and I think that's right. With the coding already done, there is only a simple circuit to build, and the Pico is a very inexpensive development board."

Nick advises us to keep our eyes open for updates to the projects in the future. Meanwhile, he's currently experimenting with using his Pico to remotely control a toy helicopter. "I'd like to teach it some new tricks, and control it by voice and/or gestures."

- Pico Pong took Nick around three weeks (in his spare time) to complete
- This is his first Pico project, but Nick has made many Raspberry Pi projects
- They include Deep Clean (**magpi.cc/deepclean**)..
- ...which watches a room and flags up which surfaces need cleaning
- His Speaker Snitch (**magpi.cc/snitch**) flashes whenever a smart speaker talks to the cloud

RUHAcam

Ruha Cheng and Penk Chen have created a printable retro-style, high-quality digital camera. **David Crookes** sees no negatives

Ruha Cheng and Penk Chen

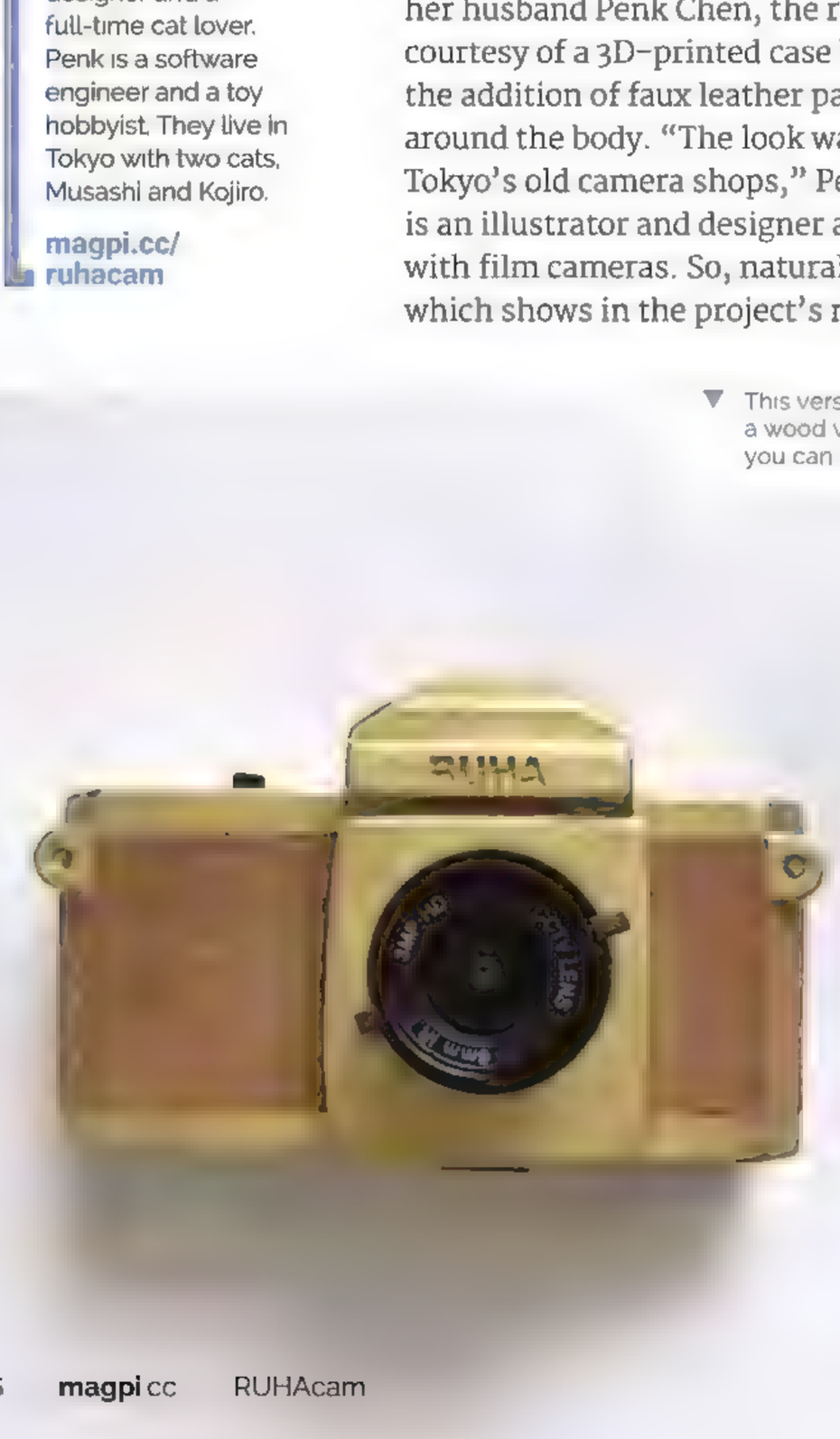
Ruha is a part-time illustrator/graphic designer and a full-time cat lover. Penk is a software engineer and a toy hobbyist. They live in Tokyo with two cats, Musashi and Kojiro.

magpi.cc/ruhacam

The High Quality Camera for Raspberry Pi has been catching the eyes of keen photographers for more than a year now. It's been placed at the centre of many wonderful projects, but seeing it form part of a camera that wouldn't have looked out of place many decades ago has made us snap to attention.

Designed by Ruha Cheng and constructed by her husband Penk Chen, the retro-styling comes courtesy of a 3D-printed case brought to life with the addition of faux leather patches wrapped around the body. "The look was inspired by Tokyo's old camera shops," Penk reveals. "Ruha is an illustrator and designer and has experience with film cameras. So, naturally she led the design, which shows in the project's name."

▼ This version of the camera includes a wood veneer look, showing how you can make the project your own



Going down the retro route has had another benefit. "The body size of an old camera is a perfect fit for DIY projects and there's lots of room for electronic parts," continues Penk, who is a software engineer by trade.

Those parts include a Raspberry Pi Zero W computer, a 2.2-inch TFT display that has been used as a viewfinder and a 2000 mAh lithium polymer (LiPo) battery. "I'm more of a smartphone camera user, but this project has been a lot of fun," Penk says.

Developing the project

From the outset the pair, who live in Tokyo, had a clear goal: to make a usable Raspberry Pi camera with easily available parts that other Raspberry Pi users could replicate. "The milestone for the project was to take a picture of the Toyko Skytree tower from our balcony," Penk says. So the couple got down to work to make it all happen.

Ruha says the easiest – and most fun – part of the build was decorating the camera. "You can choose colours and paint, and decorate it with faux leather or wood veneer," Penk adds. "The most difficult part for me was figuring out the

▼ The coup.e are working with a team from Taiwan to create an open-source camera

> The milestone for the project was to take a picture of the Toyko Skytree tower from our balcony

wiring of the LiPo battery, with a DC-DC step-up converter and USB charging module."

The decision to use Raspberry Pi Zero W was straightforward enough, though. "It has excellent balance and it's powerful enough to handle the camera's full resolution, yet small enough to fit into our 3D-printed case," says Penk. "But we opted for a smaller 2.2-inch LCD to reduce cost and skipped a video recording function so we could meet the cherry blossom."

Staying focused

Helping keep the quality high, the 12.3 megapixel High Quality Camera with its Sony IMX477 sensor supports C-and CS- mount lenses. Ruha and Penk chose a 16 mm, 10 MP telephoto lens and have been more than happy with the output.

"The image sensor size (7.9 mm) of the camera is equivalent to the sensors in the iPhone 5 era and even though it's not on a par with APS-C or M4/3 sensors commonly found in mirrorless interchangeable-lens cameras (MILC), it's definitely a luxurious upgrade from the Camera Module V2."

They've found the viewfinder helps to compose the best possible image. "It's also a lot of fun manually adjusting aperture and focal lengths, which sometimes causes 'happy accidents' that you don't get with a smartphone."

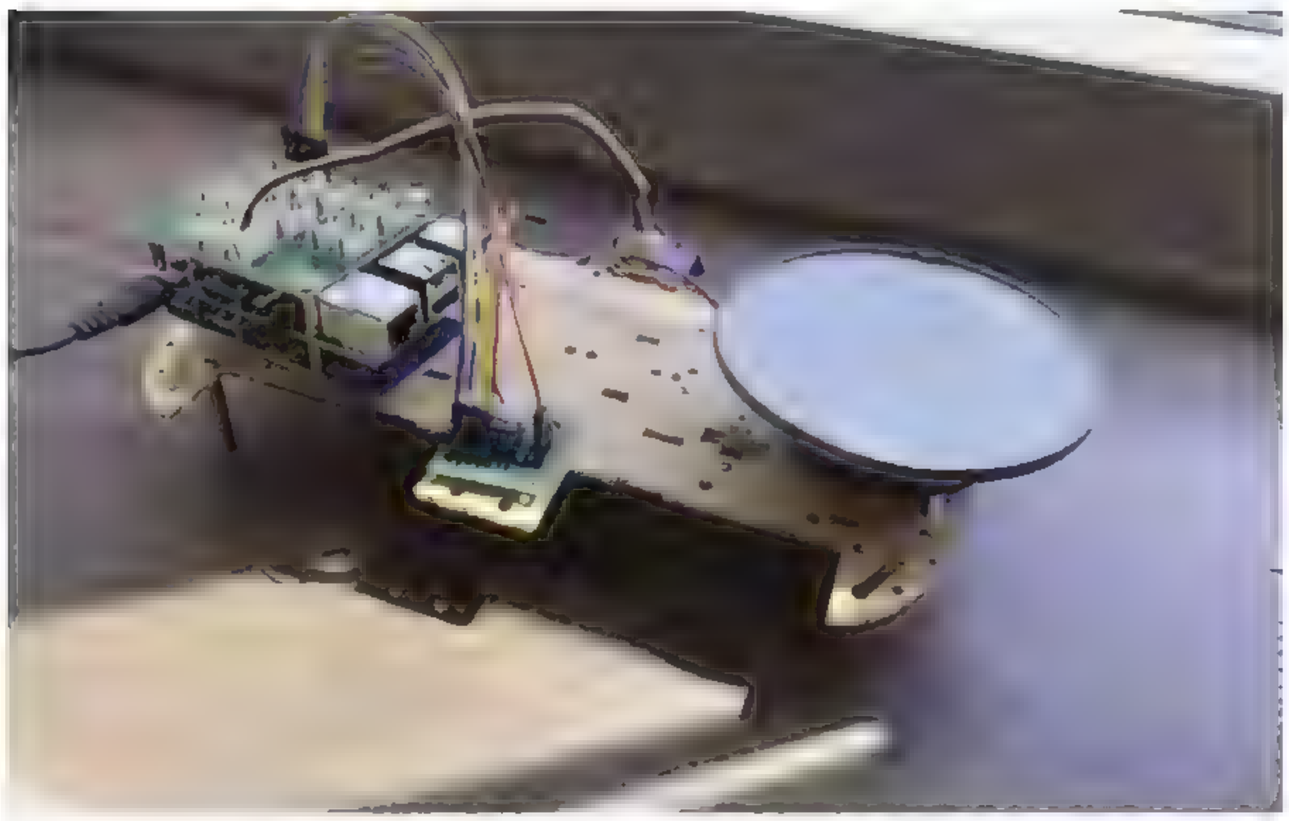
They're certainly pleased with the results. "Since the project went live, we have received a lot of interest and much feedback from enthusiastic developers," says Penk. "It's literally technology married with liberal arts."

PARTS

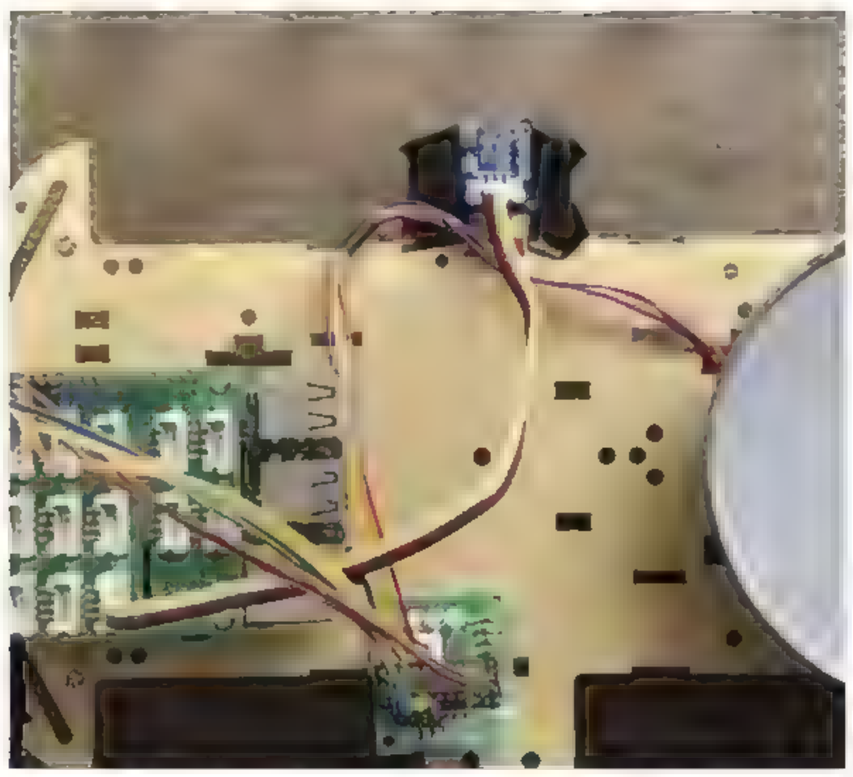
- The 3D designs are available on GitHub
- Software and design are free under MIT licence
- It uses the Raspberry Pi High Quality Camera
- Add your own design flourishes to the build
- It remains a work-in-progress

Plant Rotator

Here's a clever way to ensure indoor plants get an evenly distributed amount of sunlight. **Nicola King** dons her horticultural hat



▲ The finished project, without the plant. The blue sunlight sensor can be seen at the top



▲ A robot car chassis forms a stable base for the plant rotator

Looking after indoor plants can be challenging at times, especially when it comes to making sure that all parts of your plants are getting adequate light. This was an issue that Yohei Oka's wife was experiencing, so he decided to address the problem head-on.

"My wife was getting tired of having to turn her plant and I needed a project to work on for my 'Introduction to Making' class [at MIT]," he recalls. The bud of an idea formed and, after about a month of finessing his make, the Plant Rotator was the very practical and ingenious result.

Yohei Oka

Yohei is a first-year MBA student at MIT Sloan with a background in software engineering. He was enrolled in a super-fun class called Introduction to Making (**magpi.cc/introtomaking**), where he worked on this project.

magpi.cc/plantrotator

Innovate to rotate

The concept is simple but clever and uses sensors to monitor how much sun the plant has been subject to before rotating the 3D-printed plate that it sits on at regular intervals. "The device uses a sunlight sensor and records sunlight levels every minute," says Yohei. "Once the accumulated sunlight counter reaches a certain threshold, the device rotates the plant 90 degrees and resets the counter and repeats the same steps. If there's a lot of sunshine, it might take an hour for the plant to rotate; if there's multiple rainy days in a row, it might take a couple of days."

Yohei's six-step Instructable guide (**magpi.cc/plantrotator**) covers how he connected his Raspberry Pi 3 to a stepper motor to rotate the plant, and how he attached a Grove Sunlight Sensor. He also needed something to build the rotator onto and opted for a basic smart car chassis kit, which provides some stability. He designed the 3D-printed plate on which the plant sits. "I hadn't used CAD before taking this class, but the plate didn't take too long to design."

Yohei also needed a means of collecting and analysing the sunlight data, so he set up InfluxDB (a time series database) and Grafana (an open-source analytical application) on his Raspberry Pi. While he initially considered using an Arduino for the project, "I needed something more powerful to run cron jobs, keep track of the data collected, and visualise the data through a dashboard. I had a Raspberry Pi lying around, so decided to use that."

A great a-chive-ment

As with every project, there were obstacles to overcome. These included ensuring that the rotator was fully stable and able to rotate with the weight of the plant (around 1kg in Yohei's case), figuring out how to attach the motor shaft to the plate, and

Equipped with a Grove Base HAT for easy sensor connection, a Raspberry Pi 3 controls everything

These wires lead to a sunlight sensor; once a threshold of cumulative sunlight is reached, the plant is rotated

The plant is placed on a 3D-printed plate that sits on top of a wheel connected to a stepper motor to rotate it

establishing the threshold to determine how often the plant should be rotated. The good news is that his wife's problem plant is now "growing faster and more evenly."

Yohei's invention has already attracted lots of interest. "I've received a bunch of feedback from plant owners saying it's a great idea and want to see it productised. They've asked me if I can build one for them or create one that can support a much larger plant."

As for future iterations of his rotator, he has some plant plans. "A couple of people in my class built devices that work on auto-watering plants," he reveals. "It would be neat to combine my project with theirs so we can build a completely automated 'plant babysitting device'. It seems like plant owners would like to have such devices for when they go on longer trips."

For those of us not gifted with Monty Don's green fingers, that sounds like a blooming brilliant idea.

> My wife was getting tired of having to turn her plant

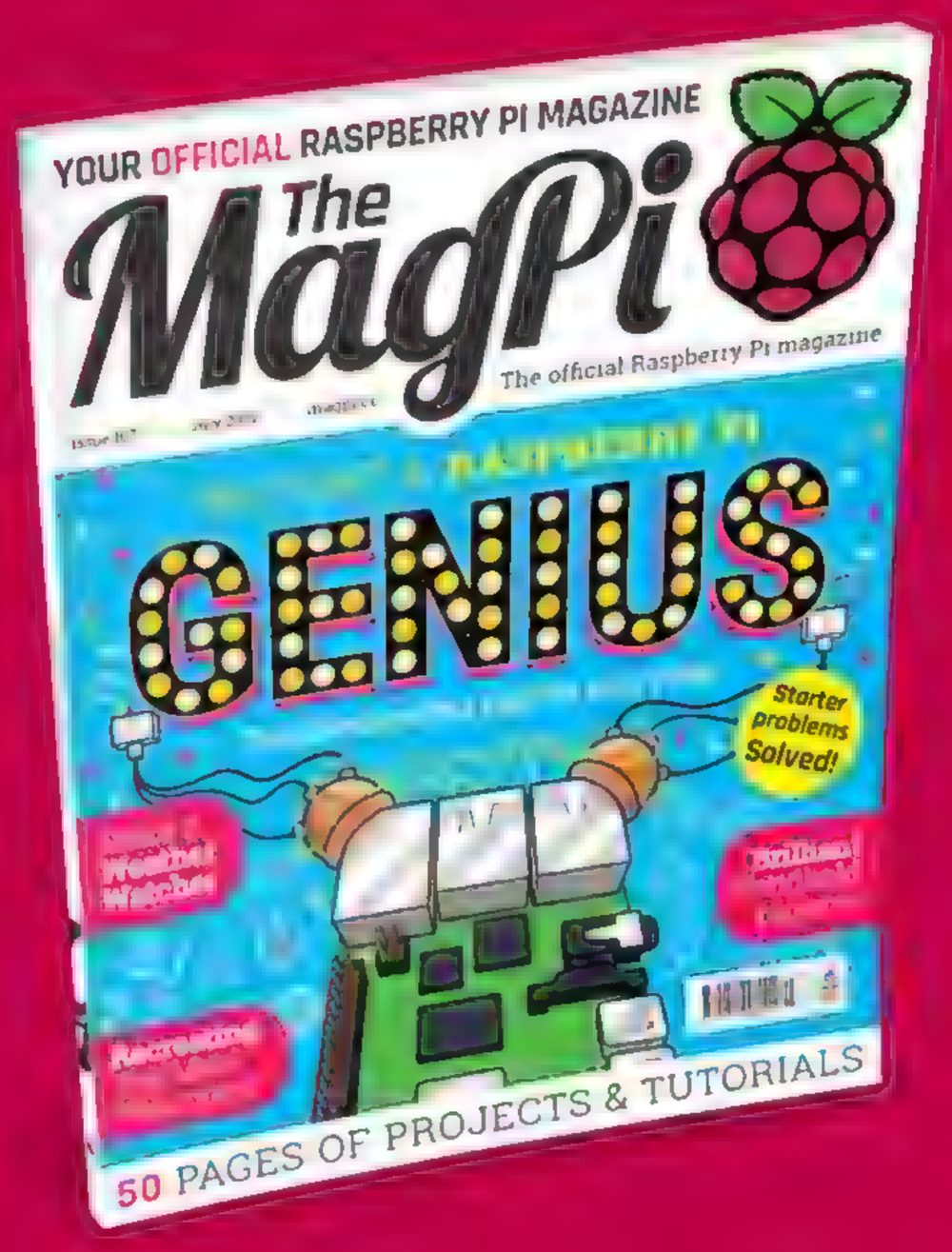
▲ Superglued to a wheel that connects to the stepper motor, the 3D-printed plate was designed by Yohei

Quick FACTS

- Yohei estimates he spent 15–20 hours on the project
- He used this tutorial to get his stepper motor working: **magpi.cc/stepperguide**
- This chassis kit was used as the base: **magpi.cc/smartchassis**
- Yohei says a big challenge was "figuring out how to be resourceful"
- He recommends testing the plant is stable on the plate before attaching everything!

SUBSCRIBE TODAY FROM ONLY £5

SAVE UP TO 35%



Subscriber Benefits

- **FREE Delivery**
  Get it fast and for FREE
- **Exclusive Offers**
  Great gifts, offers, and discounts
- **Great Savings**
  Save up to 35% compared to stores

Rolling Monthly Subscription

- Low monthly cost (from £5)
- Cancel at any time
- Free delivery to your door
- Available worldwide

Subscribe for 12 Months

£55 (UK) £90 (USA)
£80 (EU) £90 (Rest of World)

Free Raspberry Pi Zero W Kit with 12 Month upfront subscription only (no Raspberry Pi Zero W Kit with Rolling Monthly Subscription)

Subscribe by phone: **01293 312193**

Subscribe online: **magpi.cc/subscribe**

Email: **magpi@subscriptionhelpline.co.uk**

BECOME A RASPBERRY PI GENIUS

Learn to master Raspberry Pi and fix the problems that plague newbies. By **Wesley Archer** and **Lucy Hattersley**

There's nothing like the excitement of getting hold of your first Raspberry Pi, especially with the seemingly endless [illegible]. Getting to grips with it all can be a little daunting, especially for those new to the world of Linux. SSH – why are you telling me to be quiet? Sudo isn't that a board game? Headless. I want one with a head, please.

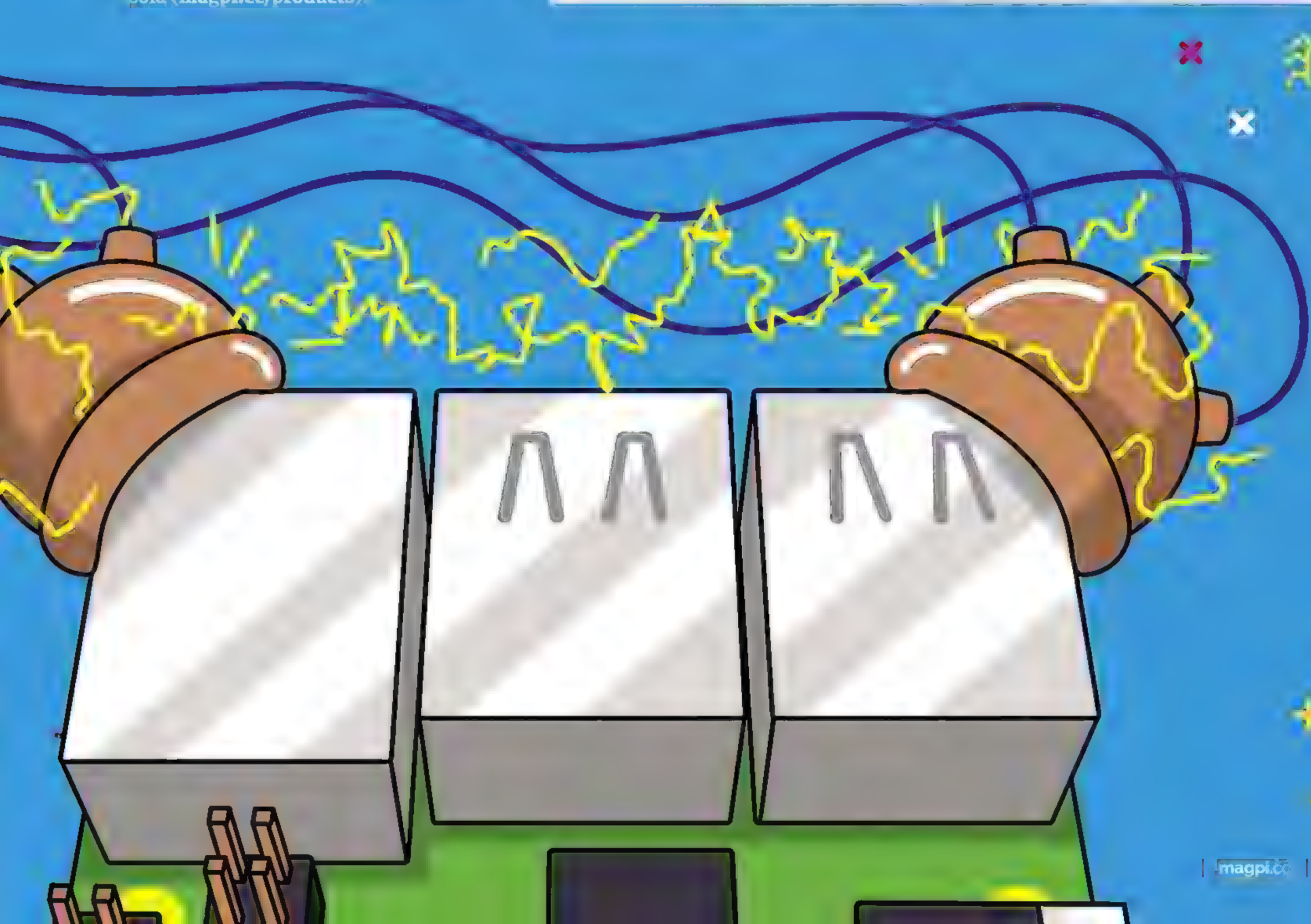
You'd be forgiven for wondering what all these new terms are, but read on and you'll not only learn what they mean, you'll become a Raspberry Pi genius in the process!

Identify your Raspberry Pi hardware

> "You should be able to identify each model of Raspberry Pi"

With a range of Raspberry Pi computers to choose from, the first step to becoming a Raspberry Pi genius is to understand the differences between each model, as each has its own strengths depending on your project. You should identify your model of Raspberry Pi before checking for help and advice.

If you want to know the precise version and revision of your Raspberry Pi, check the printed text on the board. See the Raspberry Pi product page for more information on each version sold (magpi.cc/products).



> "While it is possible your Raspberry Pi has a hardware fault, the chances are extremely slim"

Fix Raspberry Pi boot problems

This is one of the most frequently asked questions! If your Raspberry Pi refuses to boot, there are a couple of things to try.

First the good news. Your new Raspberry Pi is almost certainly not broken. Every Raspberry Pi is tested before it leaves the factory, so you can be confident that your Raspberry Pi has booted successfully at least once. While it is technically possible that your Raspberry Pi has a hardware fault, the chances are extremely slim. It's almost certainly a problem you can fix. Try these things in order:

1. You might be booting but not getting video output. First, check that your cables are connected properly. Take them all out and reconnect them. Try a different HDMI cable if you have one.

2. Try a different power supply. It's best to use an official power supply if you are in doubt – see Power Supply & Cables on Raspberry Pi's product page: **magpi.cc/products**.

3. Take out the microSD card and use Raspberry Pi Imager to flash a new installation of Raspberry Pi OS to it (see **magpi.cc/imager**).

4. Try a different microSD card. Make sure your SD card is fit-for-purpose, and not a fake. Some SD cards are unsuitable; if your card fails, it may be a good idea to try a different type. Get your microSD cards from a trusted source like The Pi Hut (**magpi.cc/sdcard**).

Raspberry Pi has full instructions for setting up your Raspberry Pi (**magpi.cc/setup**), but if it still will not boot, you will find a great post packed with advice on Raspberry Pi's forum (**magpi.cc/bootproblems**).

Understand LED warning flash codes

Long flashes	Short flashes	Status
		Generic failure to boot
		start*.elf not found
		Kernel image not found
		SDRAM failure
		Insufficient SDRAM
		In HALT state
		Partition not FAT
		Failed to read from partition
		Extended partition not FAT
		File signature/hash mismatch - Pi 4
		SPI EEPROM error - Pi 4
		SPI EEPROM is write protected - Pi 4
		Unsupported board type
		Fatal firmware error
		Power failure type A
		Power failure type B

Raspberry Pi video problems

First, check all your cables are connected and the microSD card is inserted properly. Try to re-power everything.

another computer, use:

Replacing with your Raspberry Pi IP address. You

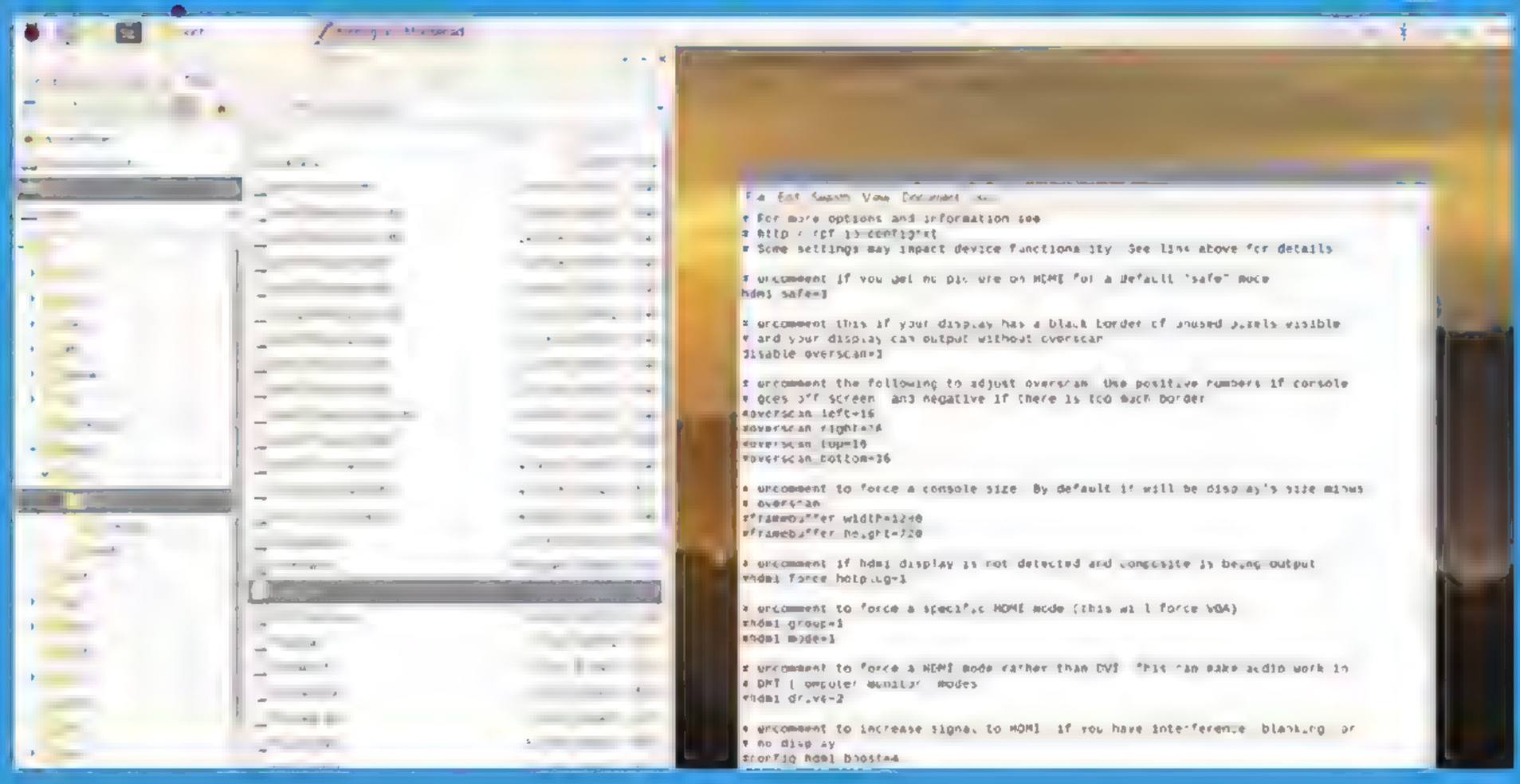
When connected via SSH, you can make changes using:

And select 2 Display Options > Default Monitor preferred resolution. Or one of the supported

SSH? In that case, remove the microSD card from Raspberry Pi and attach it to a working computer. You will be able to access a range of files in the

Raspberry Pi and boot up. See HDMI config for more information: **magpi.cc/hdmiconfig**.

> "First, check all your cables are connected and the microSD card is inserted properly"



Fix Raspberry Pi audio output

Raspberry Pi can output audio in a couple of ways – typically via HDMI or 3.5 mm jack (where applicable). Check the audio output settings and try setting the option to the way you are connected – i.e. HDMI for audio output via HDMI, typically to your display, or 3.5 mm for speakers.

Right-clicking the volume icon on the desktop taskbar brings up the audio output selector; this allows you to select between the internal audio outputs.

You can also adjust audio settings in raspi-

Select System Options > Audio and choose the option you want.

In some rare cases, you'll need to edit config.txt to force HDMI mode (instead of DVI mode, which doesn't send sound). You can do this by editing /boot/config.txt and setting `hdmi_drive=2`, then rebooting for the change to take effect. See Raspberry Pi's information (magpi.cc/audioconfig).

Use Raspberry Pi Imager

Raspberry Pi Imager (magpi.cc/imager) is used [illegible] (or another operating system). It is installed on a Windows or Mac computer, or another Linux machine (including a Raspberry Pi you might already have set up).

Prior to Imager there were other options, including NOOBS and using 'dd' on the command line. From now on, you should use Imager.

Want to know a secret code? Press CTRL+SHIFT+X when Raspberry Pi Imager is open to unlock a secret advanced options menu. Here you can preconfigure a range of options, such as enabling SSH, setting your wireless network credentials, disabling overscan, and setting a host name for Raspberry Pi.

Fix Raspberry Pi network problems

There's nothing worse than spending time setting up your Raspberry Pi, only to find out

Pi online if everything is working properly. When

Connect to a wireless LAN network using the

Wired Network icon in the upper right of the screen, and choose your wireless network from the list of SSIDs (Service Set IDentifiers). Enter the

raspi-config instead:

To scan for wireless networks, use the command:

with other useful information. Look out for the

You should be able to open up a web browser and connect to the internet, or use ping in the command line to test your connection.

But what to do if your network connection is not working properly? Here are a few steps to follow:

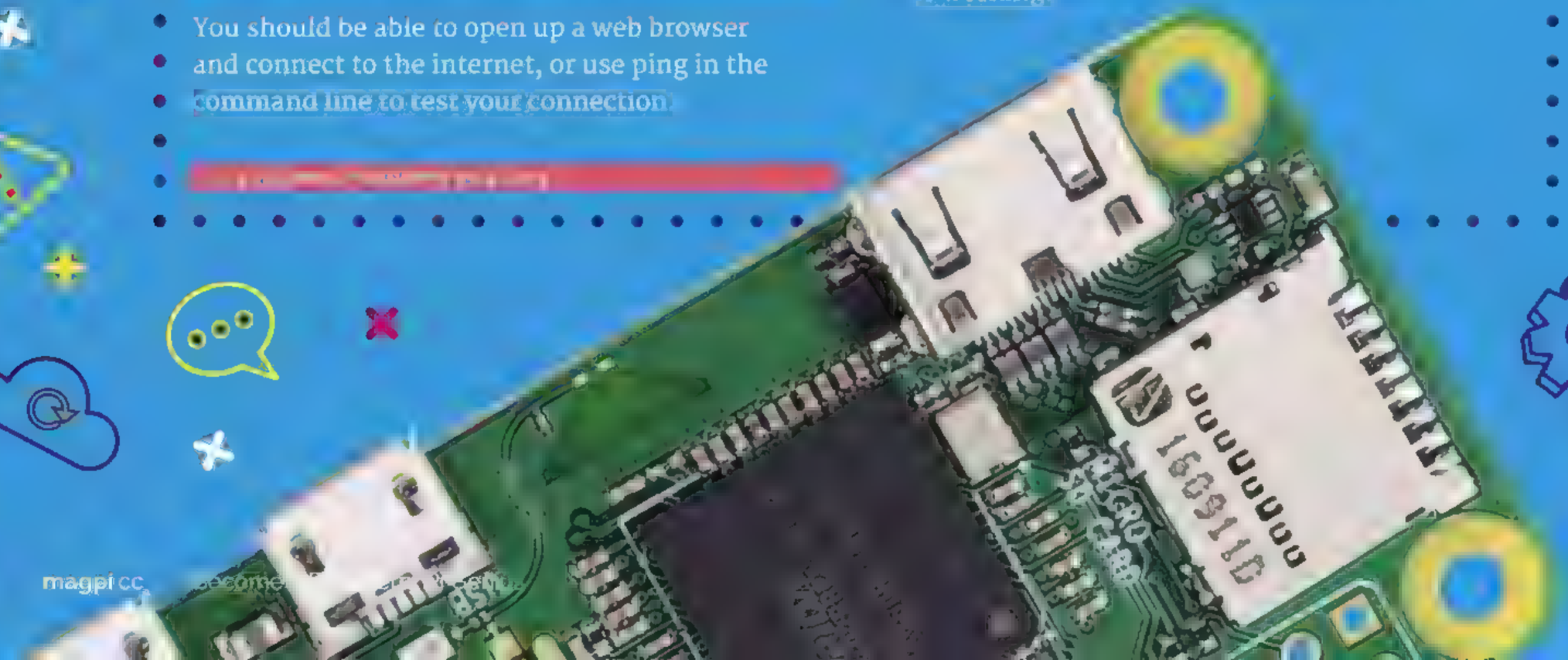
4. Check all your cables (and both ends of the Ethernet cable). Make sure it is connected

5. Check your wireless connection. If you are using a wireless LAN connection, disconnect from it and reconnect. Click on the Wired & Wireless

6. Remove and reconnect. Right-click on your connected wireless network and click OK in the Disconnect Wireless LAN alert pop-up. Head

and choose your wireless network again (and

devices and closer to the router. Try an Ethernet connection instead and make sure that it is working.

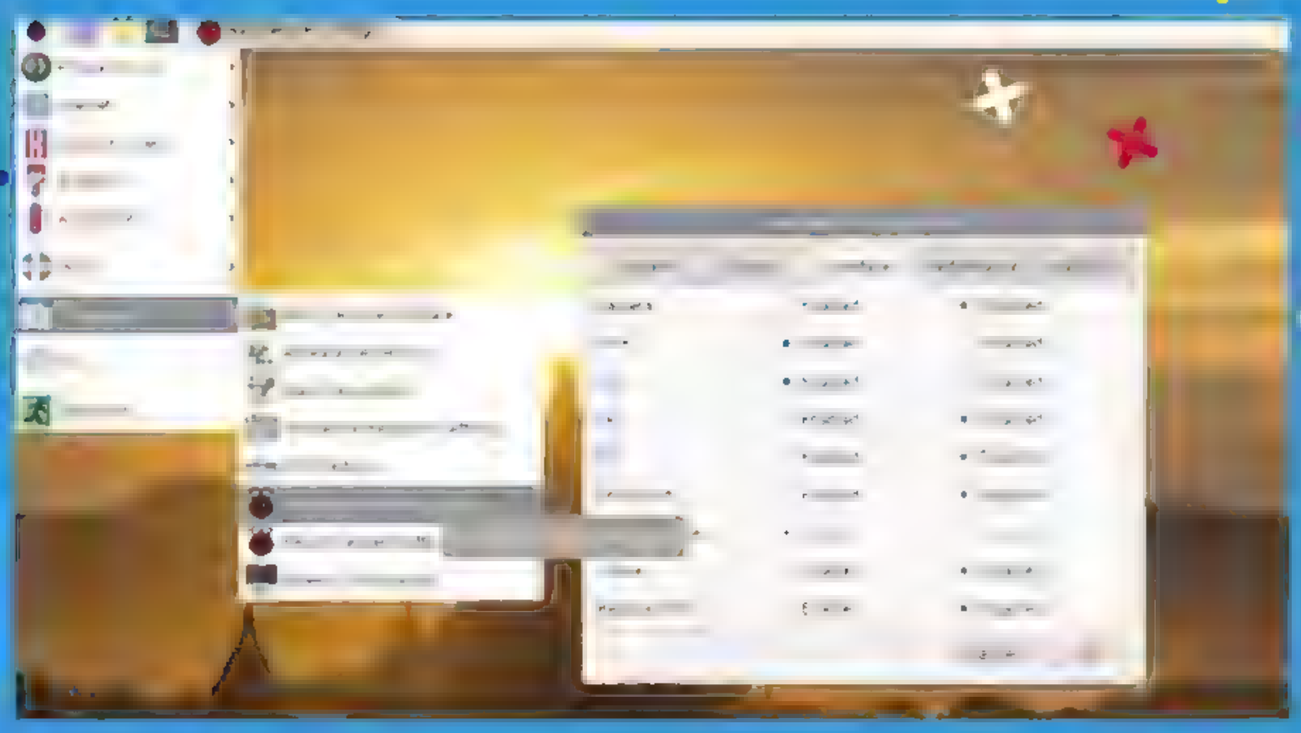


Configure Raspberry Pi

is with the Raspberry Pi Configuration tool (via the desktop). Or in the Terminal, you can use:

```
sudo raspi-config
```

Here you can fix several issues, such as enabling the Camera Module, I2C, SPI, SSH, and VNC, as well as change system options, like the audio output and changing the password for the `pi` user.



> "With wavemon, you can see other networks and what channel they are operating on"

Check your local network activity

```
sudo apt install wavemon
```

Now enter:

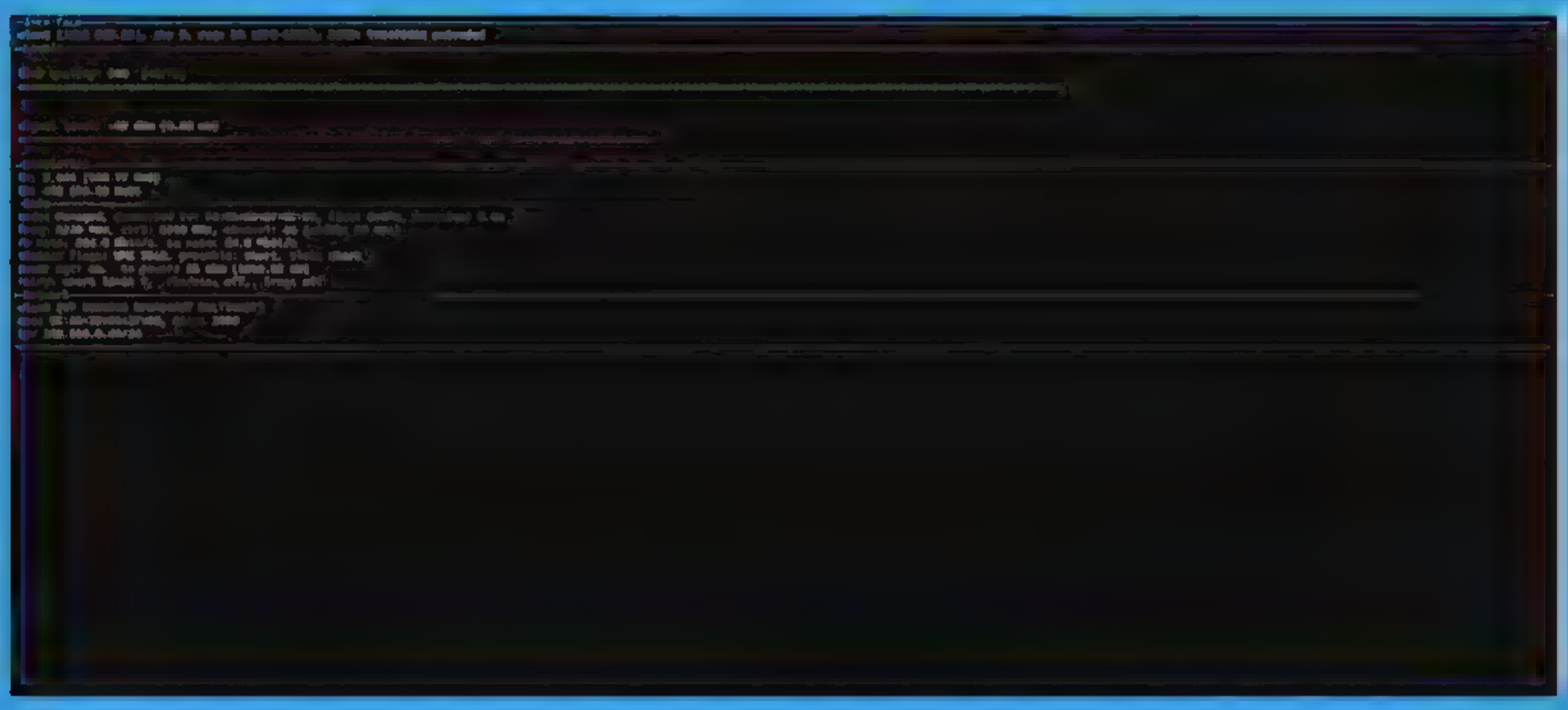
```
sudo wavemon
```

...to open a monitor for your connection. Here you can see the link quality (higher is better) and

wavemon, use the man pages:

```
man wavemon
```

With wavemon, you can see other networks and what channel they are operating on. Lots of networks in the same channel can cause interference with each other. Most routers pick a channel automatically, and you can



Going headless

"You control your Raspberry Pi remotely from another computer instead"

Get connected with SSH and VNC

References

Bookmark these pages for troubleshooting help and advice. The Raspberry Pi forum is a particularly good place to ask for help and advice.

- Raspberry Pi forum: 'Help, Resources and Technical Documents (start here)' **magpi.cc/forumhelp**
- Raspberry Pi forum: 'Getting Started' **magpi.cc/forumgetstarted**
- Raspberry Pi forum: 'Boot Problems' **magpi.cc/forumbootproblems**
- Raspberry Pi forum: 'Common pitfalls for beginners' **magpi.cc/forumpitfalls**
- Raspberry Pi Configuration documentation **magpi.cc/configuration**
- Raspberry Pi Documentation **raspberrypi.org/documentation**

Convert Raspberry Pi 4 to a tablet with RasPad 3.0

- Fully compatible with Raspberry Pi 4
- 10.1” IPS touchscreen with an impressive resolution of 1280x800 pixels
- Full access to all Raspberry Pi GPIO, Ethernet, HDMI, audio, USB, and power ports
- Compatible with Raspberry Pi OS, Arduino, Ubuntu Desktop, Chromium OS, Android and more
- 10-point multi-touch screen, 5-hour battery life
- Easy to assemble

Buy now! raspad.com/products/raspadv3

Build an arcade machine: Decorate your cabinet

You've built an arcade cabinet, but vinyl decals and edge moulding will bring it to life

K.G. Orphanides

K.G. is a writer, maker of odd games, and software preservation enthusiast. Their household can now hold very retro Street Fighter II tournaments, and that's beautiful

@KGOrphanides

Most arcade cabinet kit suppliers print pre-designed or custom vinyl decals to decorate your cabinet. Third-party printers can produce vinyls to your specification, but make sure that you provide accurate measurements.

Our vinyl decals, bought from Omniretro (**magpi.cc/omniretro**), arrived on a roll and had to be cut out, but some firms will die-cut vinyls for you. We'll use a wet application process, which makes it easier to remove and reposition decals for a short while after initial placement, to help you get a perfect alignment.

▲ Mark up in chalk pen and use a metal ruler to help cut your screen decal to size

01 Flatten your vinyl decals

If your vinyls all came on a single roll, the first step is to cut each of them out. First separate them, if they're on a single roll, but leave generous margins. Spread them out on a table or on the floor and weigh them down – coffee table books and textbooks are good for this. Leave them for at least an hour or two: 24 hours is better.

02 Cutting out

Now they're flat, it's time to cut out your vinyls. Try to get rid of all white matter on straight edges. The easiest way is to line up a long metal ruler so that it just covers the edge of the printing, and run a scalpel down the outside of it. Curved sections for the cabinet side panels are trickier, but you don't need to worry about these as they're easy to trim down once fitted. For now, trim them freehand and leave as much white overmatter as you feel comfortable with.

03 Partial disassembly

Depending on the design of your cabinet, you may need to remove a side panel to take out the acrylic marquee and screen panels. Before doing this, use a liquid chalk pen and ruler to mark the edges of your LCD display on the acrylic, so we can accurately hide the bezel.

If you've previously fitted joysticks and buttons to your control panel, this is the time to remove them too. Apply steady pressure to the rear of snap-in style buttons to pop them out of the cabinet. People with large fingers may find a ButterCade Snap Out Tool useful for this.

04 Applying vinyl to your marquee acrylic

Two acrylic parts require individual application of vinyls: the marquee and the screen that goes in front of your monitor. The former is easy: remove the backing from the vinyl marquee decal and any protective film from the acrylic. Spray both the acrylic and the adhesive back of the vinyl with two or three squirts of application fluid. You want them to be damp all over but not awash.

Pick up the vinyl decal in both hands and, starting at one end of the acrylic, line it up with the edges and paste it down. If you're not happy with the positioning, firmly hold the vinyl and snap it back up – the application fluid will help it release easily

Once it's positioned, use your applicator and a cloth to smooth it down, drive out any excess water, and remove any trapped air bubbles under the vinyl. Trim any excess vinyl spilling off the edge of the acrylic with a knife.

05 Measuring your screen acrylic

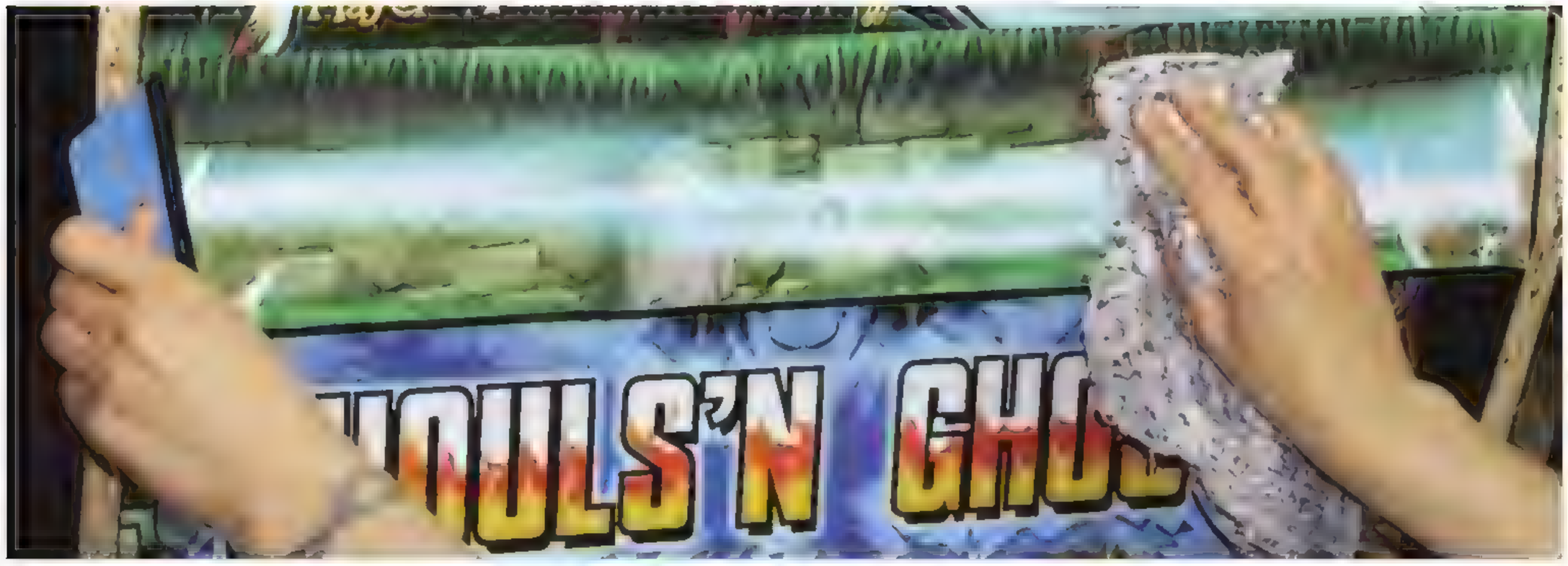
Cutting your screen decal to size is awkward. Before removing the screen acrylic from the cab, we marked the inner position of our monitor's bezel on the acrylic using a chalk pen. If your cabinet has a detachable VESA mount, bring the monitor with you to help line everything up.

Grab your screen vinyl and mark up the area to cut out

Measure the distance between the edge of the acrylic and the chalk line you drew on it. Measure in multiple places to be sure of distances. Our 24-inch monitor's positioning and bezel size means that we cut 35 mm in at the top and sides, and 65 mm from the bottom – yours will differ.

06 Cutting your screen decal

Once you've taken the measurements, grab your screen vinyl and mark up the area to cut out. Mark on the side showing the picture, paying particular care to the corner positions. Double-check these by placing the acrylic on top to make sure both sets of marks line up.



▲ Use a vinyl applicator and a cloth to stick down, remove excess moisture, and eliminate air bubbles from your decals

Grab your metal ruler, place it along your marked line, and cut a rectangle out of the middle of the vinyl decal with a blade. If in doubt, err towards leaving too much vinyl rather than too little. To check positioning, put the acrylic over your monitor, and your vinyl over the acrylic: they should all line up.

You'll Need

- Vinyl decals
- U-moulding/ T-moulding
- Scalpels/craft knives
- Strong scissors
- Liquid chalk marker pen
- Metal rulers, tape measures
- Vinyl application fluid
- Vinyl applicator
- Neoprene glue

07 Screen decal application

Now, turn the vinyl upside down, remove its backing, spray it and the acrylic with the application solution, and stick it down using an applicator and cloth. Residual chalk marks can be wiped off using a bit more of the application solution.

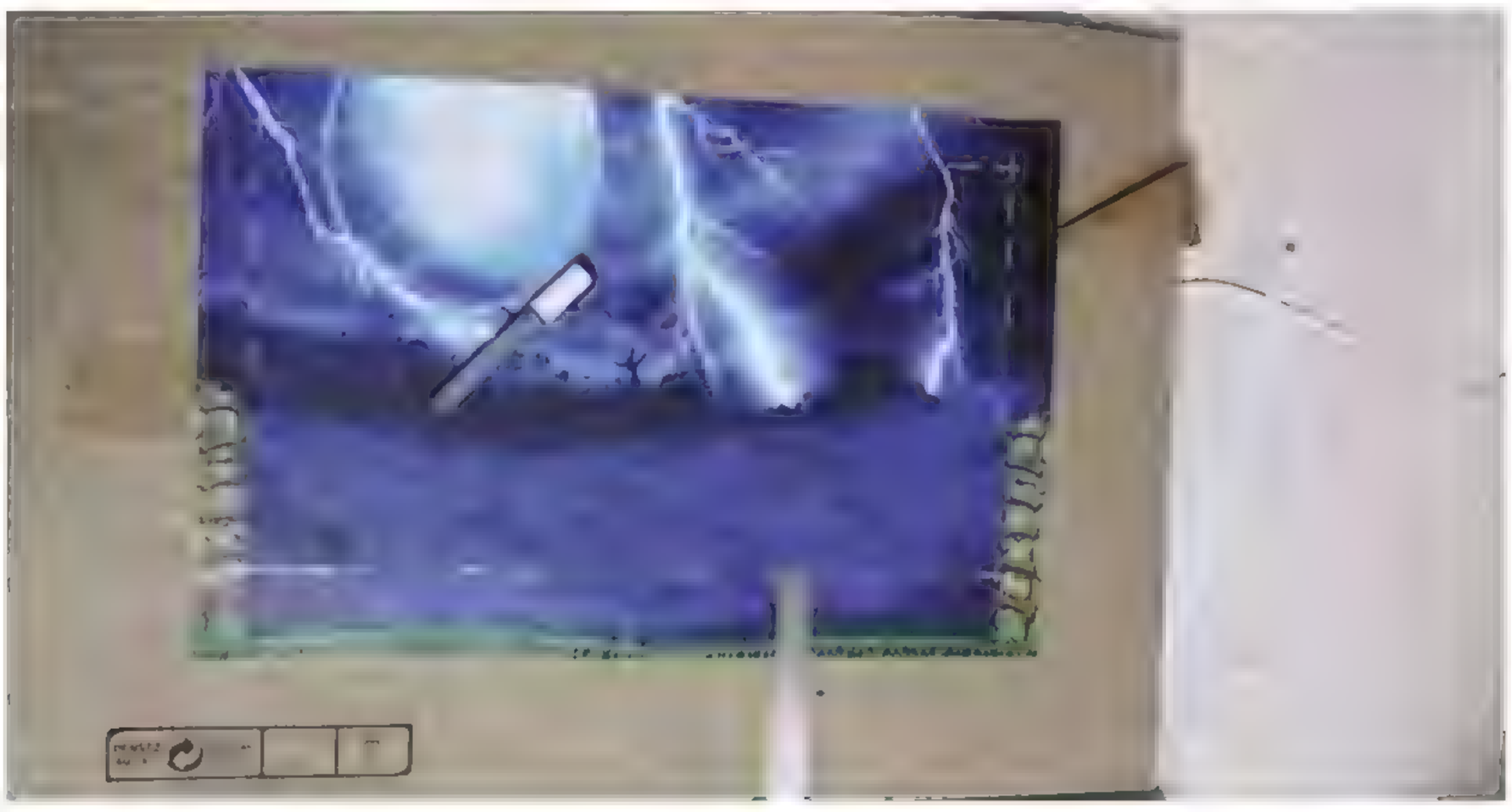
Allow both the marquee and screen decals to dry for a day, trim them if needed, slide them back into your cabinet, and reattach anything you removed. This will probably be the last time you do this, so make sure the side panels are on securely and are correctly lined up and bolted to your stand, if you have one.

If you plan on back-lighting your marquee, this is a good time to put in your light. We used adhesive tape and supplied clips to mount a 50 cm USB-powered LED light on the underside of the marquee, just in front of the speakers.

08 Applying flat vinyls

If you have a full-height cabinet or a bartop and stand, you'll probably have a number of flat, front-facing areas to decorate – in our case, the front cupboard door of our stand, its base, and the front of its foot. Do these next to get your hand in.

The drill is the same for all of them: place the vinyl decal face-down on the floor, remove its backing, spray both it and the surface you're



▲ We marked the inner position of our monitor's bezel on the acrylic using a chalk pen

applying it to, position your decal, and smooth it out with your applicator. Use a scalpel to trim off any overmatter. For the door, we applied the decal with the door in place – knob removed, starting at the top. We had to open the door to flatten and trim the vinyl in places.

09 Control panel decals

Most control panel decals wrap around the top and front of your panel. Buttons and joysticks should not be present during application. This is a relatively easy section to apply, but watch your position if there are decorative patterns designed to surround specific buttons or joysticks.

You may need to trim overmatter from the sides with a scalpel to get the decal to fold over the front face properly. Be careful when smoothing the vinyl on this fold, as it can be prone to both trapped air bubbles and damage from the join beneath.

10 Cabinet positioning

Side panels are the largest pieces of vinyl you'll be applying, but they're less intimidating than they seem. For a standalone bartop, one person can mount them in a vertical position with little fuss, as shown in Omniretro's video at **magpi.cc/omniretrovinyl**.

Full-height cabinets present more of a challenge due to their height and the size of the vinyl – a second person is useful here. You can apply long vinyls in an upright position, but we'd already attached rubber feet to our cabinet, so we used these to help pivot the cab down to lie on a sheet of cardboard on the floor.

11 Apply side panel vinyls

Lying flat and sprayed down as before, it's easy to line up the side-panel decal. Make sure everything's covered – with two people, it's easy to snap the decal back up if you make a mistake, then use a cloth and applicator to drive out excess moisture. Use a Stanley knife to trim the vinyl to size – its solid metal body makes it easy to follow the line of the cabinet's curves.

Go around again to remove any air bubbles and ideally leave the vinyl to dry for at least a couple of hours before pivoting the cabinet back up and lowering it to expose the opposite side. Repeat the process.

You can leave some white-space overmatter on side panel decals before application, as they're easy to trim with a knife afterwards

> "Side panels are the largest pieces of vinyl you'll be applying"

If your cabinet has separate stand and bartop parts, but uses a single sticker, there will be a slight ridge where these join. However, careful application (and a sympathetic vinyl design) makes this effectively invisible. Just be careful smoothing around it.

Top Tip

Screen materials

Acrylic scratches really easily, so tinted tempered glass is an excellent alternative for your cabinet screen.

Highly flexible, U- and T-moulding are used to give a clean finish to the cabinet edges

Demonstrated here without glue flex U-moulding backwards and use a finger or thumb to press it into place on a cabinet edge

Warning!
Solvents

Always use solvents in a well-ventilated area and keep away from open flames

magpi.cc/solvents

12 Moulding

We used U-moulding on our cabinet, with neoprene glue to hold it in place securely. First, measure and use scissors to cut two strips to go above and below the marquee – it's better to cut these a few millimetres too long and then trim than it is to have a gap.

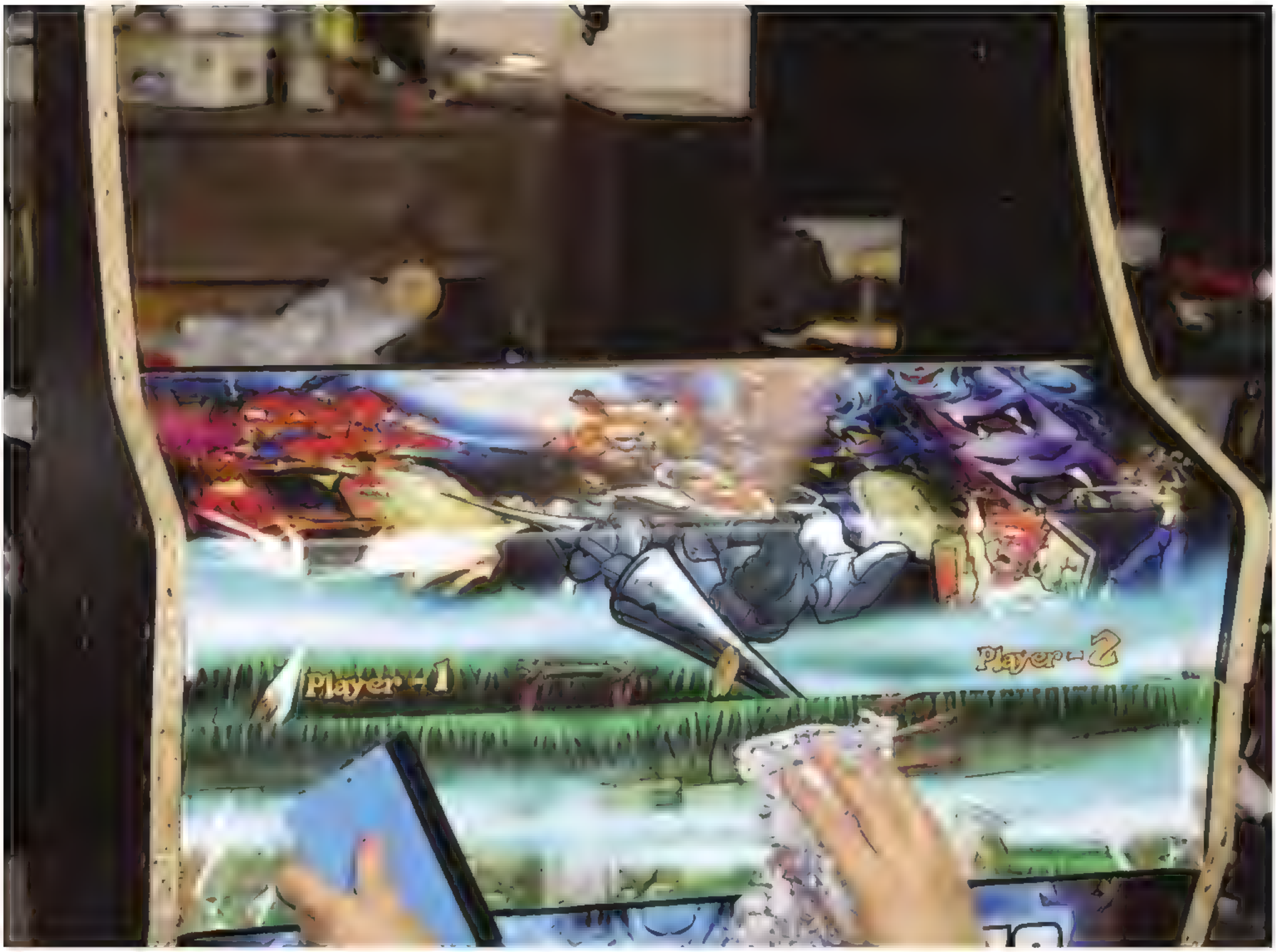
Use a spatula to help apply neoprene glue along the edge you're working on, then use the tube's nozzle to apply glue to the inside of the U-moulding.

To lock U-moulding into place, bend it backwards to spread the U-shaped section, push that onto the edge you're applying it to, and then roll the moulding down along the edge, using a finger to push it into place.

When applying it to a long section, such as each side of your cabinet, start at the front underside – rubber feet help access here – apply glue to the cabinet edge and the first 50 cm of your roll of moulding, and have someone else feed it to you as you work up and around the cabinet. When you get to the bottom at the back, cut off your moulding with scissors.

T-moulding locks into a pre-cut groove along the edges of your cabinet, making it more secure, but it's still a good idea to apply glue to the flat surfaces for security. Either way, use a rubber mallet to gently tap down your moulding at the end.

You can use acetone to clean the glue off your hands and the moulding, but keep it away from the vinyl.



After spraying the vinyl decal, and the acrylic, with our homemade fluid, we applied it and smoothed down with an applicator and cloth

Vinyl application fluid

You can buy commercial vinyl application fluid (**magpi.cc/vinylfluid**), widely used by car customisation enthusiasts to apply decals, but we filled a spray bottle with the following homemade formula:

- 66 ml surgical spirit
- 132 ml water
- 2 drops washing up liquid

You can use warm water with a drop of washing up liquid alone, but the surgical spirit reduces drying times, which means less waiting between different stages of application and decorating.

12 Finishing moves

Use a scalpel to cut out the vinyl above the button holes: locate a hole, pierce it with the blade, slice until you find the edge of the hole, and then follow the hole round to remove all the vinyl. Do this for all your joystick and button holes.

As described in *The MagPi* #105 (**magpi.cc/105**), screw your joysticks back into place from the inside. If you're going to put protective acrylic panels over your control panel, this is the time to do it – they're held on solely by the buttons.

However, because our cabinet is for home use, we've left the vinyl bare for a more comfortable and attractive finish. If your cabinet will see lots of play, acrylic will protect it and cut down on wear and tear. Whichever you choose, connect a DuPont cable to each button and pop them into place.

Follow the instructions from issue 106 to connect your buttons and peripherals to Raspberry Pi.

Warning!
Sharp objects

Take care when using knives and scalpels.

magpi.cc/handknives

Build a Weather Watcher station with public data

Don't rely on last night's forecast. With Raspberry Pi Zero and an e-ink display, you can track weather changes in real-time

Nik Rawlinson

Pencil and paint fiddler who believes the best things in life begin with 'c': cats, coffee, and camper-vans.

nikrawlinson.com

You're heading out: what do you take? Hat, gloves, and scarf, or shorts, shades, and sun cream? The only way to be sure is to check the weather before you leave. With this project, which uses a 4-inch e-ink panel and Raspberry Pi, you can set up a weather display by your front door. Weather Watcher draws down live data from the web and updates every quarter of an hour – or more often if you choose. You get all the benefits of your own weather station without the additional expense or effort involved in taking readings from your own garden.

01 Attach your screen

We're going to display our data on a 4-inch Inky wHAT screen. It uses e-ink, which we've chosen because it's easy to read in bright sunlight, has wide viewing angles, and holds the image without drawing power between each refresh. We've chosen the black and red version, which is slightly more expensive than the mono edition, but as this project works just as well in all-black, you can opt for the mono version instead if you're working on a budget. With your Raspberry Pi switched off, attach the panel to the GPIO header and secure it using the included standoff screws.

02 Install the display

Boot your Raspberry Pi. We're using Raspberry Pi Zero W with a pre-soldered header for this project as we'll only be performing one action every 15 minutes and speed isn't important. When it's finished booting, either open a Terminal window if you're using the graphical interface, or connect remotely using SSH. At the prompt, type:

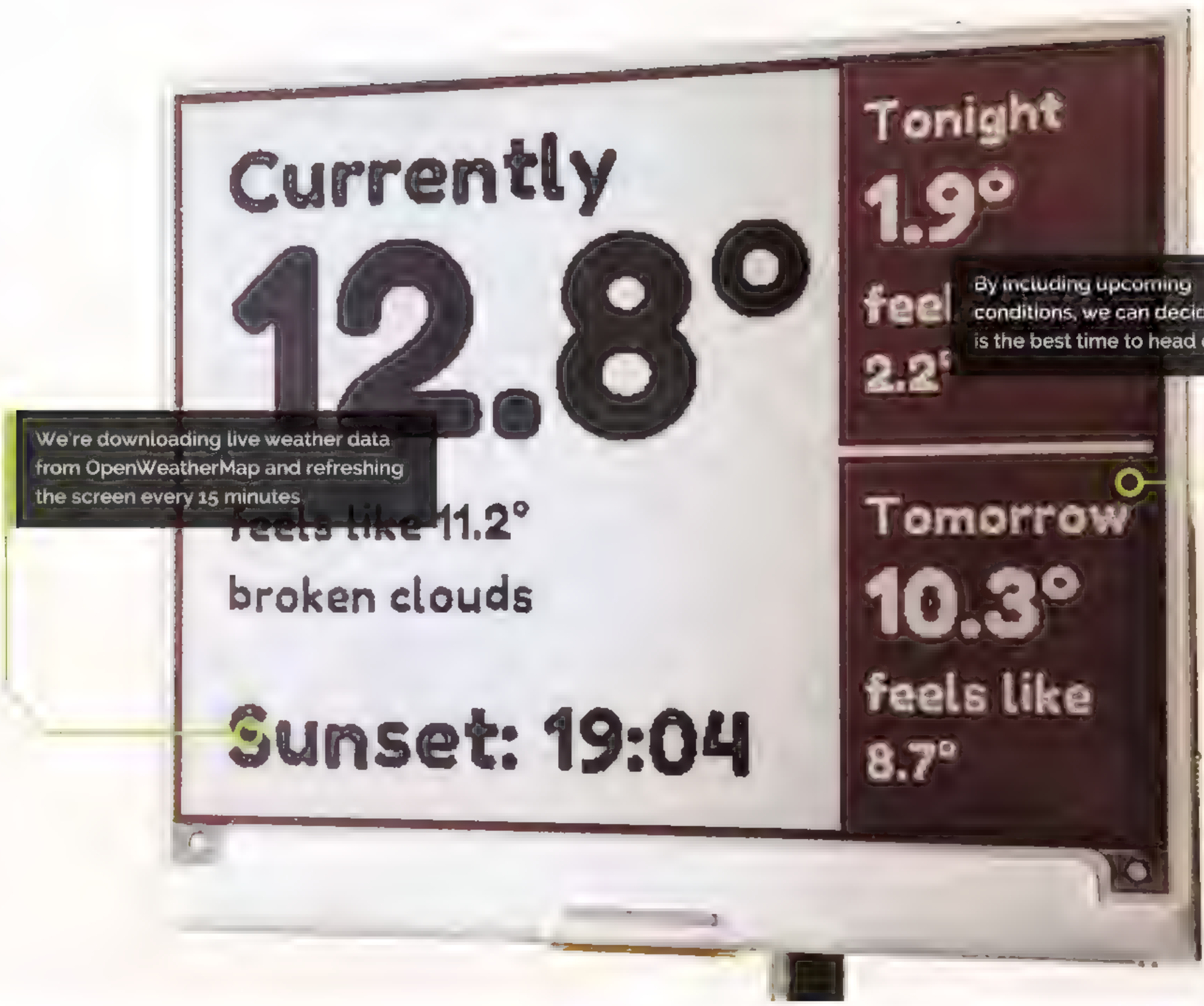
```
curl https://get.pimoroni.com/inky | bash
```

Press **ENTER** and wait for the display's drivers, fonts, and example code to install. This can take several minutes on Raspberry Pi Zero. When it's finished, reboot your Raspberry Pi to make sure everything's been recognised.

03 Grab an API key

We're going to draw down live weather conditions from OpenWeatherMap, which very generously makes a lot of its data available for free. To make sure it can tie each query to

If you don't have the room or budget for a 4-inch display like the one we're using here, the code could be adapted for smaller panels



a specific user (so it can check it's not being abused), you need to sign up for a free API key, which must be passed to the server every time you refresh the feed. Point your web browser at **home.openweathermap.org/users/sign_up** and provide the required details (make sure you're happy with the terms, conditions, and policies before completing the sign-up). Verify your email address by clicking the link it sends to your inbox.

04 Copy your API key

Return to **home.openweathermap.org**, log in, and click 'API keys'. You can generate several keys for different applications if you choose, and OpenWeatherMap will combine their cumulative usage to make sure you don't exceed your free account limits (which are 1000 calls a day or 30,000 calls a month). You'll be making fewer than 150 calls a day for this project and, for the moment, can manage with a single API key, so copy the long string of digits and letters in the default Key box and save it in a text document. You'll need it later.

05 Set up PyOWM

The OpenWeatherMap data you'll be using is delivered in indexed arrays, from which you can extract individual variables using Python. However, someone has already done the hard work for us by developing the PyOWM library, which bundles together your API key and weather query and submits them to the OpenWeatherMap servers. To install the library, type the following at the Terminal prompt:

```
pip3 install pyowm
```

06 Download and edit our code

Everything is in place. You now just need to customise our code for your geographical location. Download the **weather-panel.py** script from **magpi.cc/weathergit** and save it at the location on your Raspberry Pi from which you'll run it. For simplicity, we've saved it in the root of our user folder, at **/home/pi**. If you're using the graphical interface and you've never worked with Python

You'll Need

- Inky wHAT e-ink display **magpi.cc/inkywhat**
- OpenWeatherMap API key **magpi.cc/weatherapi**
- PyOWM library **magpi.cc/pyowm**

Top Tip

Faster refreshing

Even if you have a colour display, switching `"red"` on line 42 for `"black"` will speed up the refresh rate, so long as you're happy with mono output.

code before, open it in Thonny Python IDE, which you'll find in the Programming submenu. If you prefer to stick with the terminal prompt, try Nano, by navigating to the folder containing the script and typing:

```
nano weather-panel.py
```

07 Define your forecast location

Move your cursor down to lines 10 and 11 and replace `[lat]` and `[long]` with the co-ordinates for the location for which you want to show the weather. If you don't know what these are, switch to your browser and visit **latlong.net**. Type in a location, being as specific as you can to avoid picking somewhere with the same name elsewhere in the world. When the site centres its maps on your chosen location, click within the satellite view on the exact spot you want if the indicator isn't quite on target. Copy and paste the co-ordinates below the map and satellite windows into your code.

08 Provide your API key

Paste your API on line 13, using it to replace `[your API key]` (keep the quotes in your code but get rid of the square brackets). Assigning the key to a variable lets the PyOWM library roll it into each request it makes, effectively unlocking your account in the process. The remainder of the block immediately below the API line pulls down temperatures for today, tonight, and tomorrow, and tonight's sunset time, all of which can be useful if you're deciding how long you can put off mowing the lawn.

09 Get your colours right

Lines 55 to 69 handle positioning the data on the screen, using black text for the largest part of the display, which shows the current conditions, and white text on a solid background for the smaller text showing what it will be like tonight and tomorrow. Drawing the solid background is handled by line 52, which defines a single large rectangle

Gooseneck mini USB cables are ideal for mounting your Raspberry Pi and screen close to your front door. Just make sure the plug remains firmly in place

```
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) an$
#
# m h  dom mon dow   command

@reboot python3 /home/pi/weather-panel.py &
*/15 0,6-23 * * * sudo python3 /home/pi/weather-panel.py &

^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify
```

Crontab can be used to invoke jobs at set intervals. Here, we're using it to run the weather-panel.py code every 15 minutes

starting on the top row, 280 pixels across the screen, and extending to the bottom-right corner. Specifying `fill = 2` blocks it in with red pixels. If we had a mono panel, we'd use `fill = 1` for black. Line 53 then draws a four-pixel-deep white line across the middle of the rectangle to divide it into two boxes.

10 Test your weather panel

Save your tweaked code (if you're using Nano, do this by pressing **CTRL+X** to quit, and press **Y** when Nano asks if you want to save). Now, at the Terminal prompt, populate your e-ink panel for the first time by running the code. From the same folder in which your code is saved, type:

```
python3 ./weather-panel.py
```

The process will take a few seconds to initiate as it first needs to request the data from OpenWeatherMap, then extract the variables you require, and arrange the screen contents before physically refreshing the panel. Don't assume something's wrong if nothing seems to be happening at first.

11 Set your panel to auto-refresh

Did it work? Good. Now you need to set it to refresh every 15 minutes (or more often if you prefer). Type:

```
sudo crontab -e
```

If you've never edited crontab before, you'll be asked which editor you want to use. Pick Nano (option 1). Key down to the bottom of the file and add the following two lines:

```
@reboot python3 /home/pi/weather-panel.py &
*/15 0,6-23 * * * python3 /home/pi/weather-panel.py
```

Press **CTRL+X** to quit Nano, and press **Y** when asked if you want to save.

12 Understanding that schedule

The first of those two lines tells your Raspberry Pi to run the script as soon as it has rebooted. The second line also tells it to run that script, but every 15 minutes between 6am and midnight (there's no point stressing the screen when we're in bed). The first `*/15` effectively translates to 'every 15th minute' – i.e. every quarter. The `0,6-23` signifies 'at zero hour (midnight), then every hour between 06:00 and 23:00'. The three asterisks that follow apply those timings every day, month, and day of the week, while the `&` on the `@reboot` line makes the script run in the background so it doesn't hold up the rest of the boot process.

Pico-Voice

Use Raspberry Pi Pico to make your own voice processing and sound effects system

Mike Cook

Veteran magazine author from the old days, writer of the Body Build series plus co-author of *Raspberry Pi for Dummies*, *Raspberry Pi Projects*, and *Raspberry Pi Projects for Dummies*.

magpi.cc/mikecook

You'll Need

- Microphone headset **magpi.cc/headsetpb**
- Pico Voice hardware **magpi.cc/106**

Last month we saw how to make the hardware for the Pico-Voice changer. This month we breathe life into it by showing how to make ten funky effects using software.

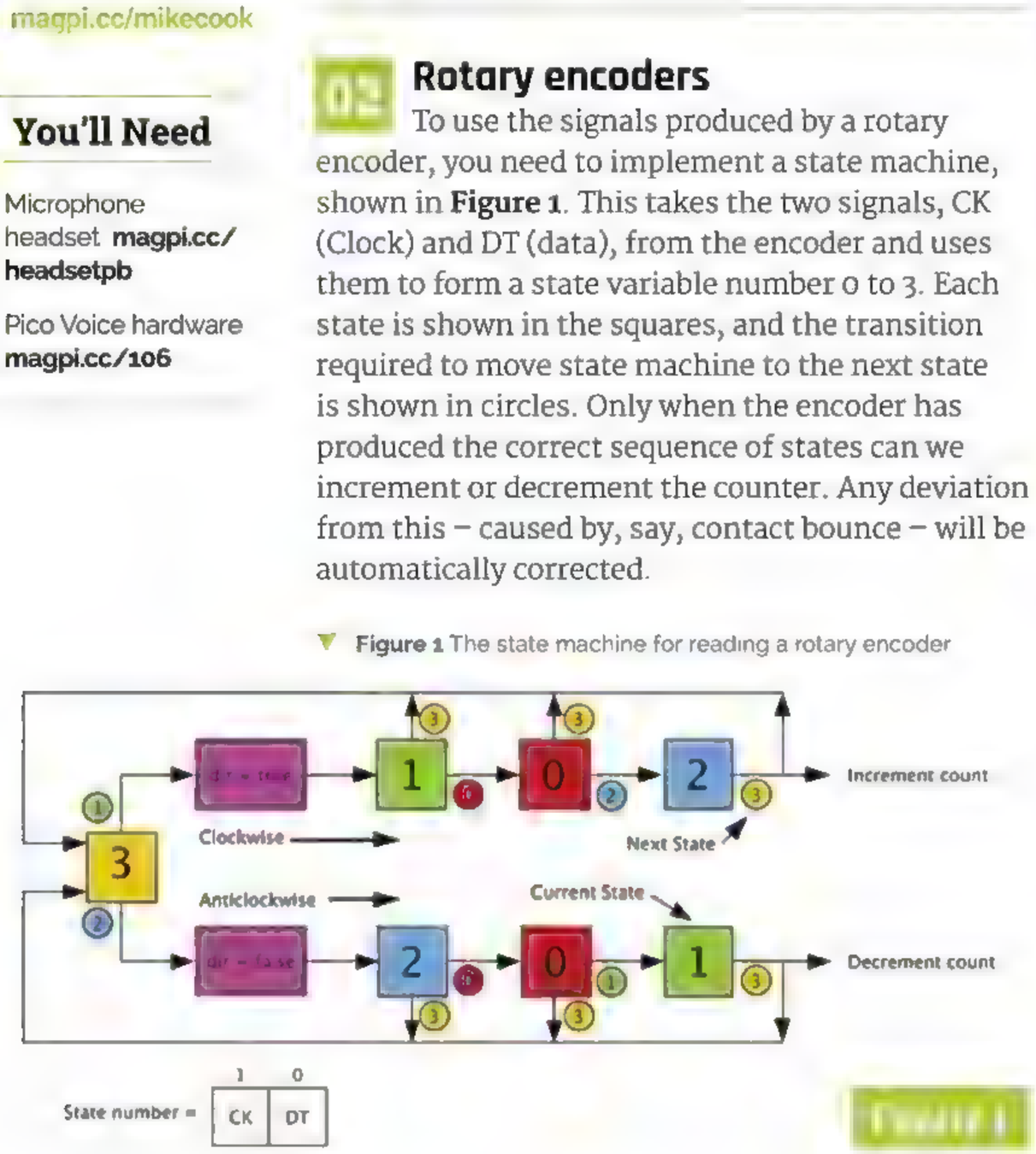
01 Infrastructure code

As Pico is relatively new, there aren't always the support libraries you need. The first problem we faced was the lack of support for the OLED display. There was a C++ library at **magpi.cc/picossoled**, but we couldn't get it to work using just plain C. So we asked on the Raspberry Pi forum, and the owner of this GitHub site, Martin Kooij, stepped forward and reworked his solution to make it compatible with C. Many thanks to him for that. We also had to handle the rotary encoders in C, as the previous solution we used had been in Python.

02 Rotary encoders

To use the signals produced by a rotary encoder, you need to implement a state machine, shown in **Figure 1**. This takes the two signals, CK (Clock) and DT (data), from the encoder and uses them to form a state variable number 0 to 3. Each state is shown in the squares, and the transition required to move state machine to the next state is shown in circles. Only when the encoder has produced the correct sequence of states can we increment or decrement the counter. Any deviation from this – caused by, say, contact bounce – will be automatically corrected.

▼ **Figure 1** The state machine for reading a rotary encoder



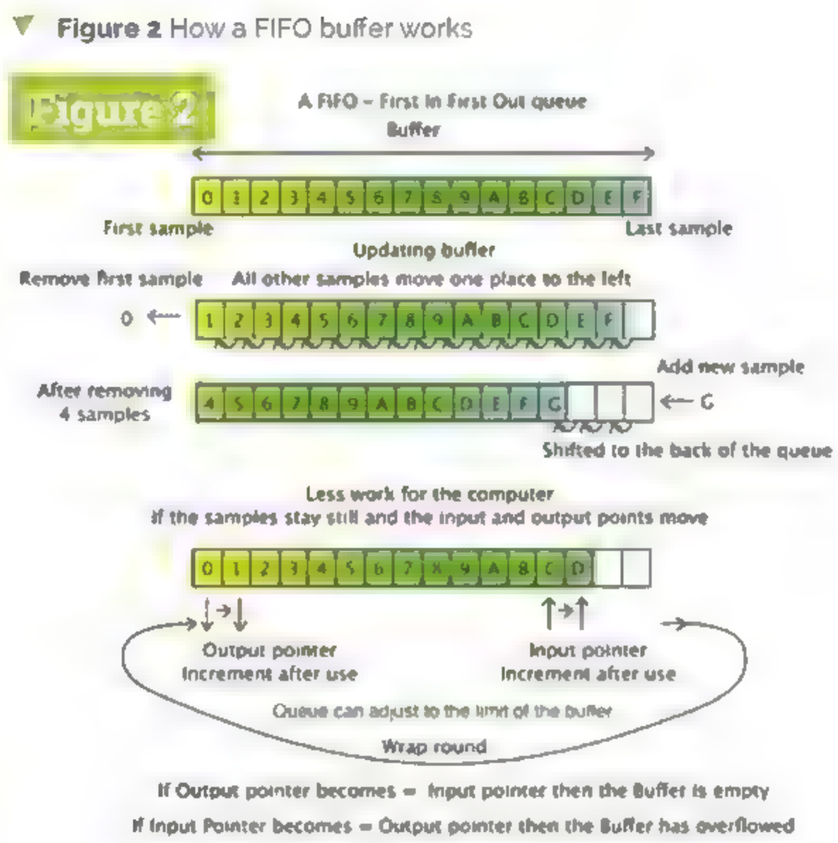
03 FIFO buffer

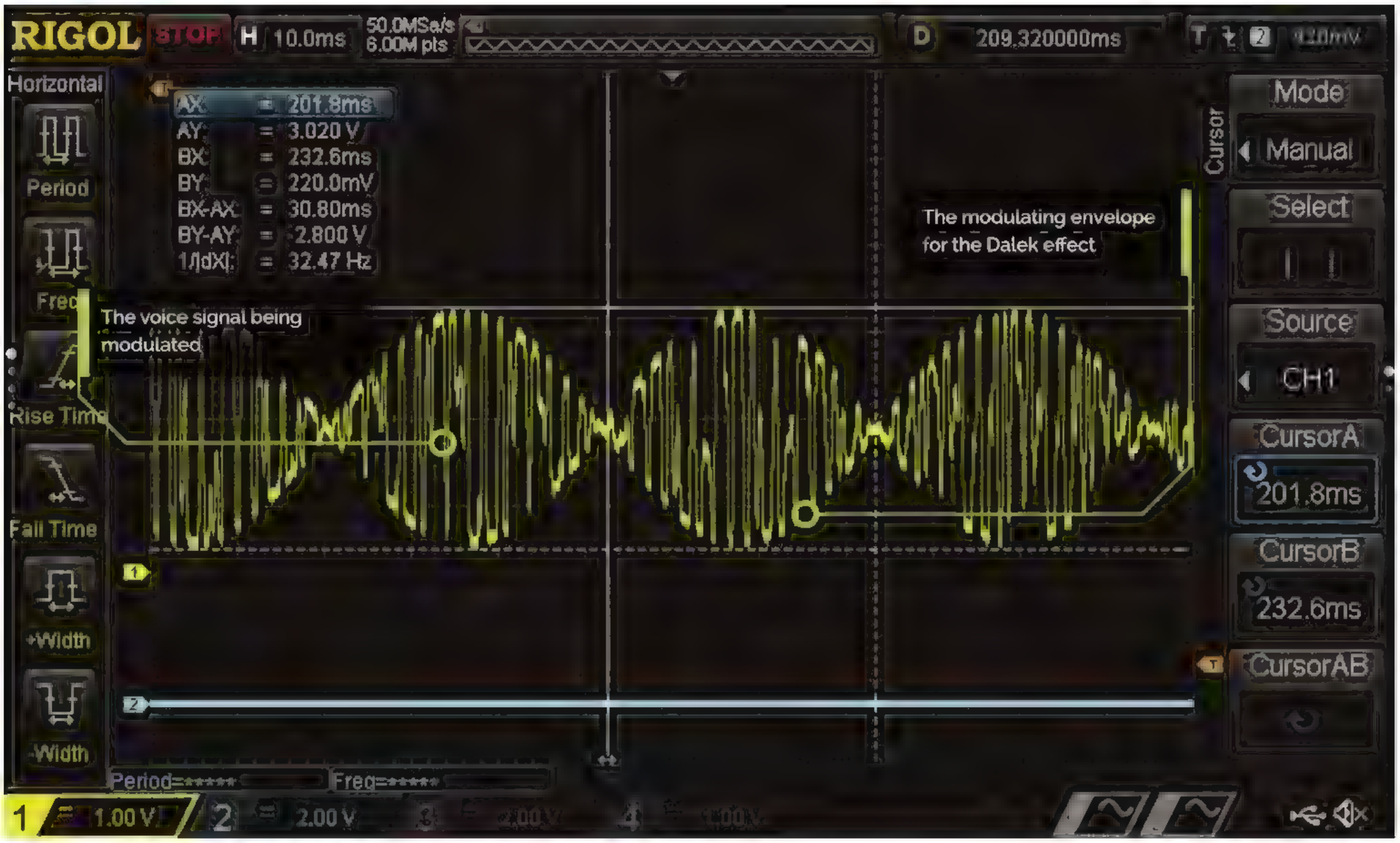
A FIFO is a First In, First Out buffer or queue, and is the basis of all but one of the effects. **Figure 2** shows how they work: just like a post office queue, the first to arrive is the first who is dealt with. However, rather than have the data shuffled along in memory, it is far quicker if you use the concept of pointers to say which place in memory you add new arrivals and which place you read the data to process. By keeping the input and output pointer moving at the same rate, you can implement a delay.

04 Effect 0 - Delay

A diagram describing a delay is shown in **Figure 3**. Samples are taken from the voice signal at a regular interval – in the case of most effects here, this is at a rate of 20,000 samples per second – and placed in the buffer. Then a sample is removed from the buffer and sent to the DAC (digital-to-analogue converter), and both pointers incremented. They are then checked to see if they have exceeded the buffer length and set back to zero if they have. Most people have considerable difficulty in talking when they hear their own voice coming back half a second later.

▼ **Figure 2** How a FIFO buffer works





Effects 1 and 2 – Reverberation and Echo

These might be thought of as different effects, but are implemented with the same code, only the setup parameters are different. These two effects are shown in **Figure 4**, overleaf. Unlike the delay, there are two or more output pointers – in our case, five. These are distributed throughout the buffer to give five increasing delays. Each one has a fixed attenuation before being passed on to an attenuation controlled by one rotary encoder. These delayed signals, together with the current input sample, are added together to give the output. The placing of the buffer taps and the fixed attenuation differ in the two effects.

> The trick is to have a very short buffer of a few milliseconds

in **Figure 5**. The trick is to have a very short buffer of a few milliseconds, and step the output buffer at a fractional rate. Of course, the buffer only has integer memory locations, so when accessing the memory you take the integer part of the output pointer, but the pointer itself and the increment value are floating-point variables. We thought it would be fun to have two pointers picking up different parts of the waveform and applying a variable shift to each.

Top Tip

Head set

You will need a headset with separate jack plugs for microphone and headphones, not a USB connection.

Effects 3 and 4 – Pitch Shift and Two Voices

Shifting the pitch up or down is not as difficult as it might sound – the basic arrangement is shown

Effects 5 and 6 – Backwards

If you take a simple delay but drive the output pointer backwards – that is, decrement

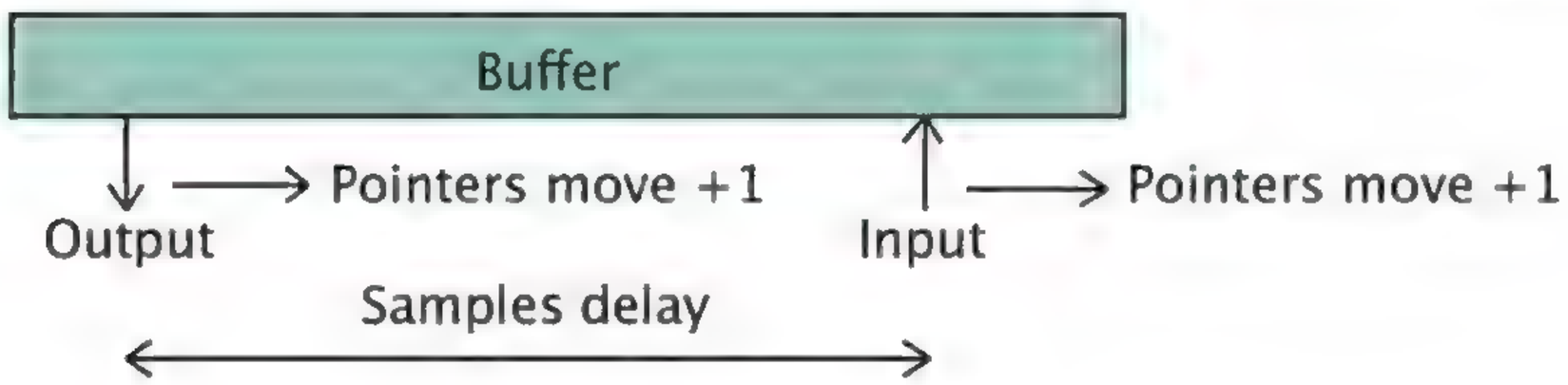
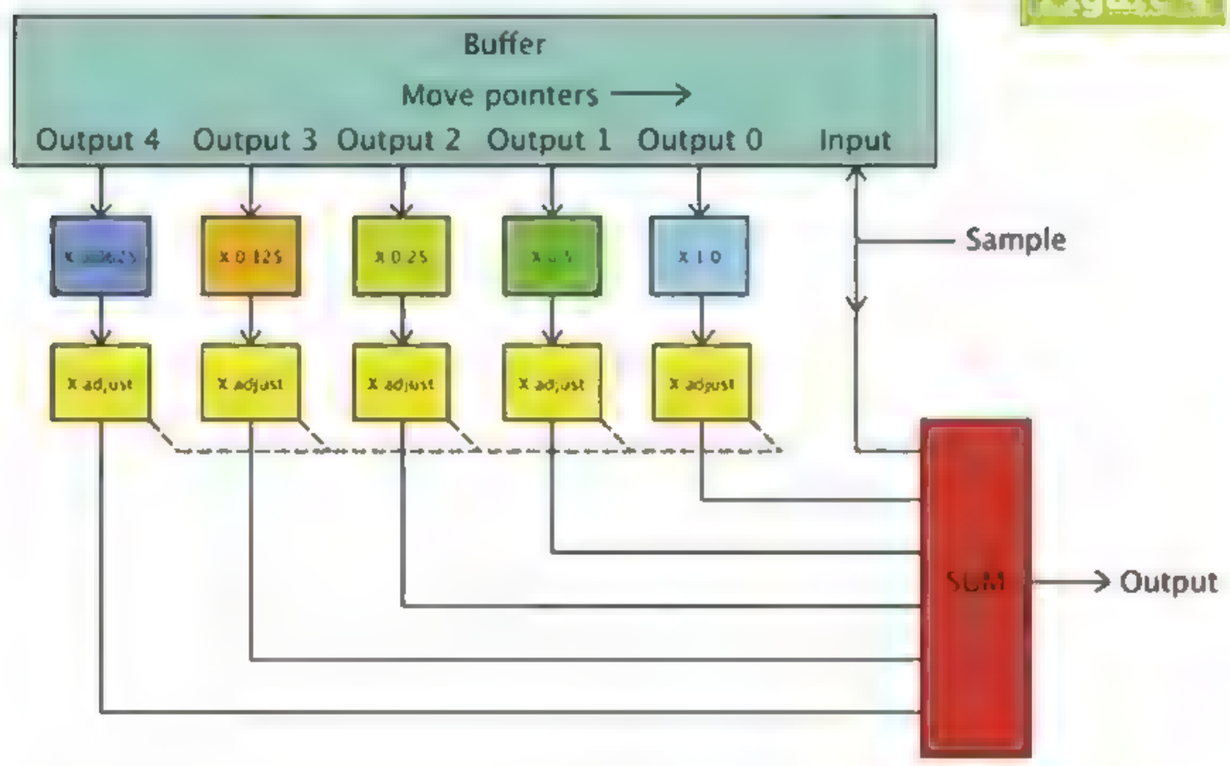
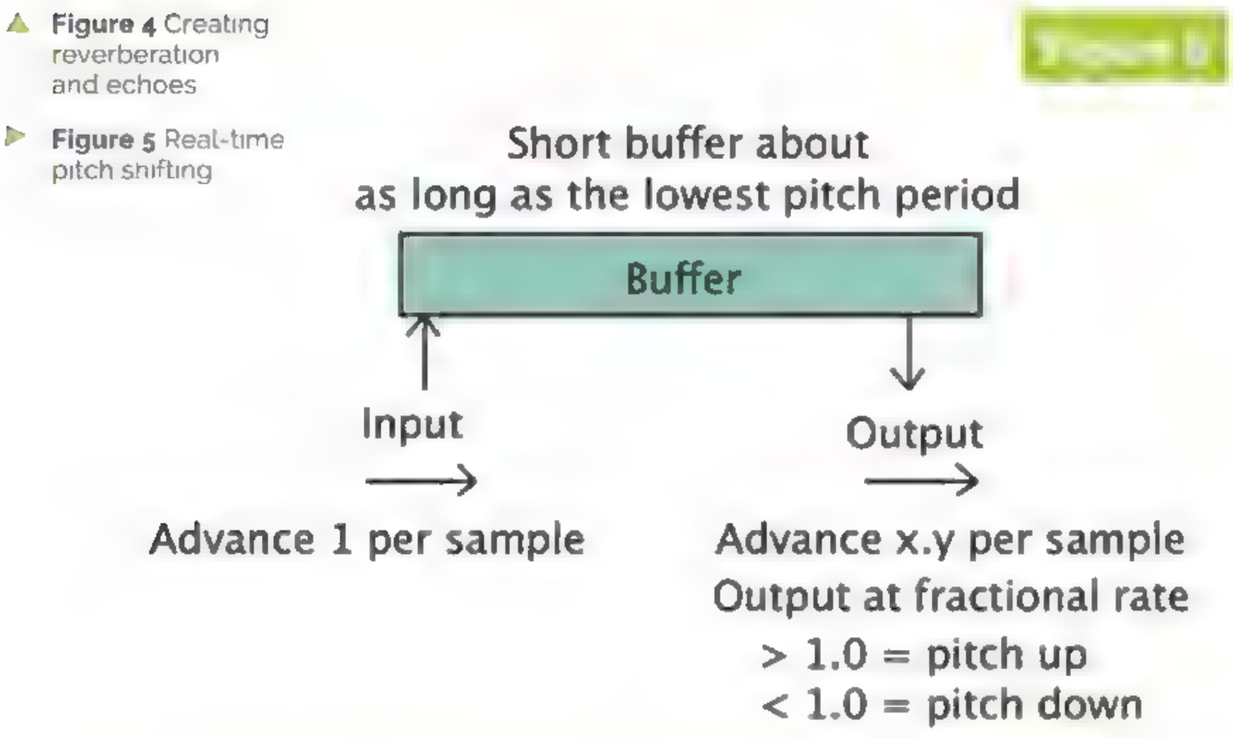


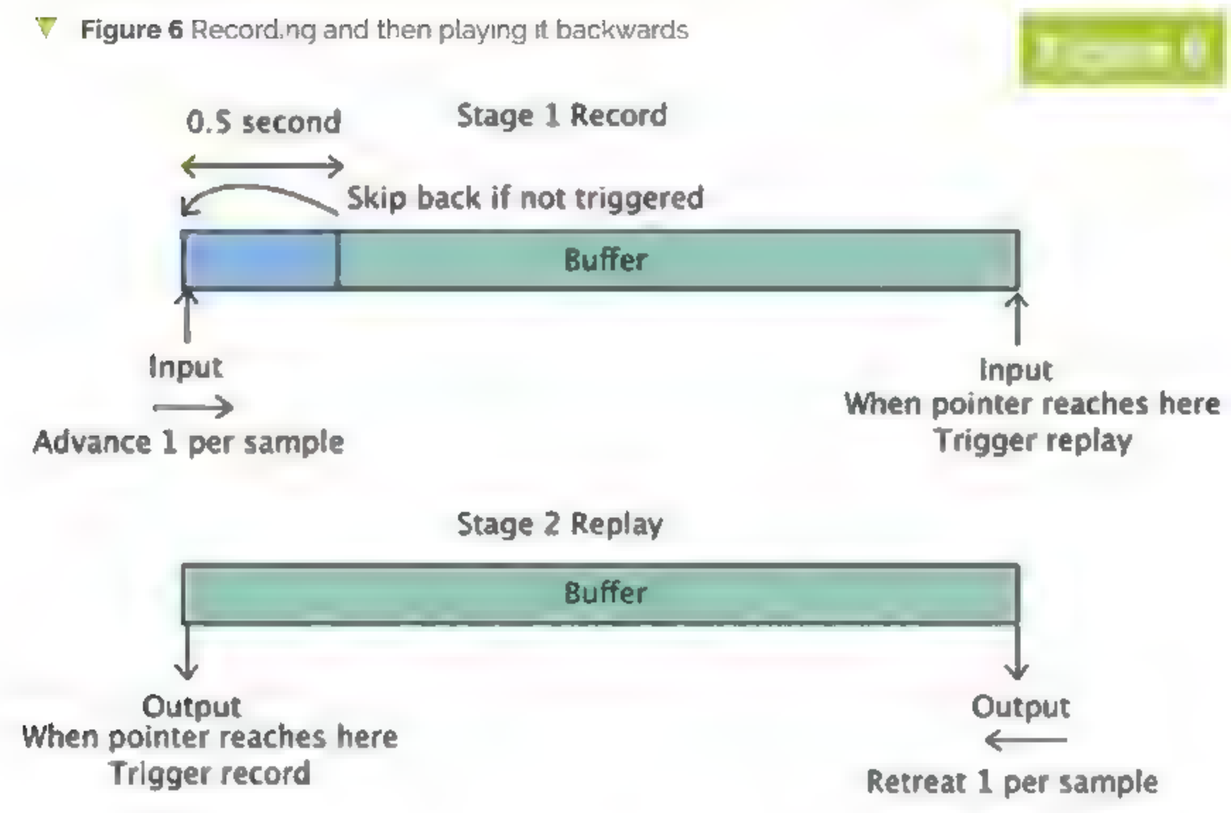
Figure 3 Implementing a simple delay



▲ **Figure 4** Creating reverberation and echoes



▶ **Figure 5** Real-time pitch shifting



▼ **Figure 6** Recording and then playing it backwards

it – the result is that it is played backwards, and sounds like you are talking in Russian. You actually get a variable delay, but you need that to generate the real-time backwards effect. It is great fun to try to say things backwards and hear them come out recognisably forwards, although sometimes it sounds a bit slurred. **Figure 6** shows a record/replay version of the backwards effect, which is great for learning individual words. You can set a trigger level so you don't start recording until a sound is made.

Effect 7 – A Dalek

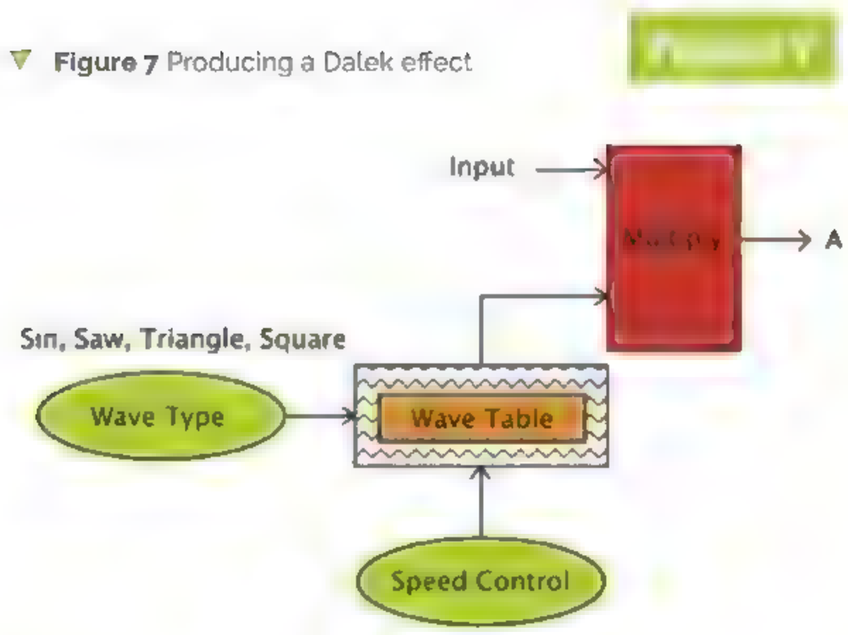
When the BBC created the Daleks for *Doctor Who*, they asked their Radiophonic Workshop to create a voice for them. This was done by using a ring modulator to treat a voice, so its amplitude was modulated by a sine wave. In other words, the volume of the sound was turned up and down rapidly, about 30 times a second; this is known as amplitude modulation, or tremolo. This creates both the sum and difference of the two signals being multiplied together. **Figure 7** shows how we created it. The square and saw-tooth waves sound gritty due to the sudden jump in the waveform.

Effect 8 – Vibrato or a Wobbulator

While modulating the amplitude of a signal is called tremolo, modulating the frequency of a signal is known as vibrato. This is our favourite effect and is great fun. It makes the voice wobble, and controlling the rate of the wobble can make some very silly sounds. We use the same basic technique as the pitch shift effects, but with the increment rate of the output pointer varying by the output of a sine wave generator. This is shown in **Figure 8**, and the depth of the effect and the speed of the wobble are controlled by the rotary encoders.

Effect 9 – Musique concrète

Musique concrète is a French phrase describing a technique of recording sounds on tape, chopping it up, and joining the bits together in a random sequence. The theoretical concepts were first outlined the 1940s, but it became popular in the 1950s with the advent of tape recorders. For our effect, we record a buffer of sound samples and then continuously play back sections of sound in a random order. Pushing the right-hand rotary encoder stops the playback and puts the system into record mode again. **Figure 9** shows how we do this with the buffer split into four sections.



▼ **Figure 7** Producing a Dalek effect

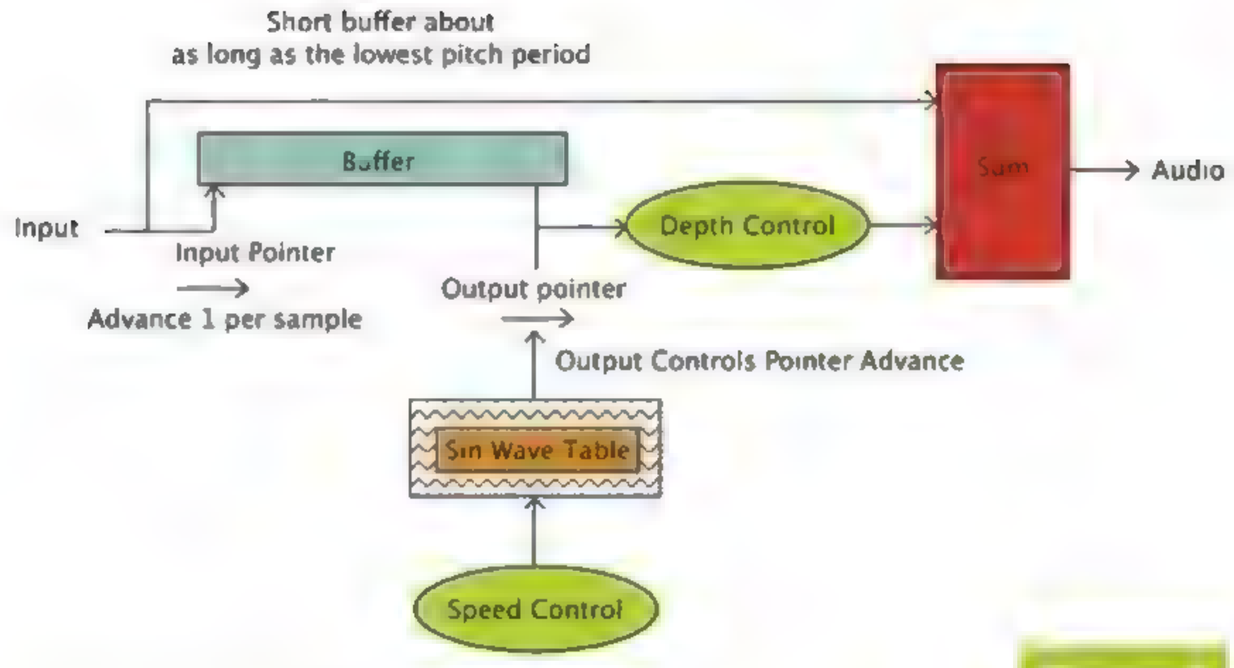
> Modulating the frequency of a signal is known as vibrato

10 The software

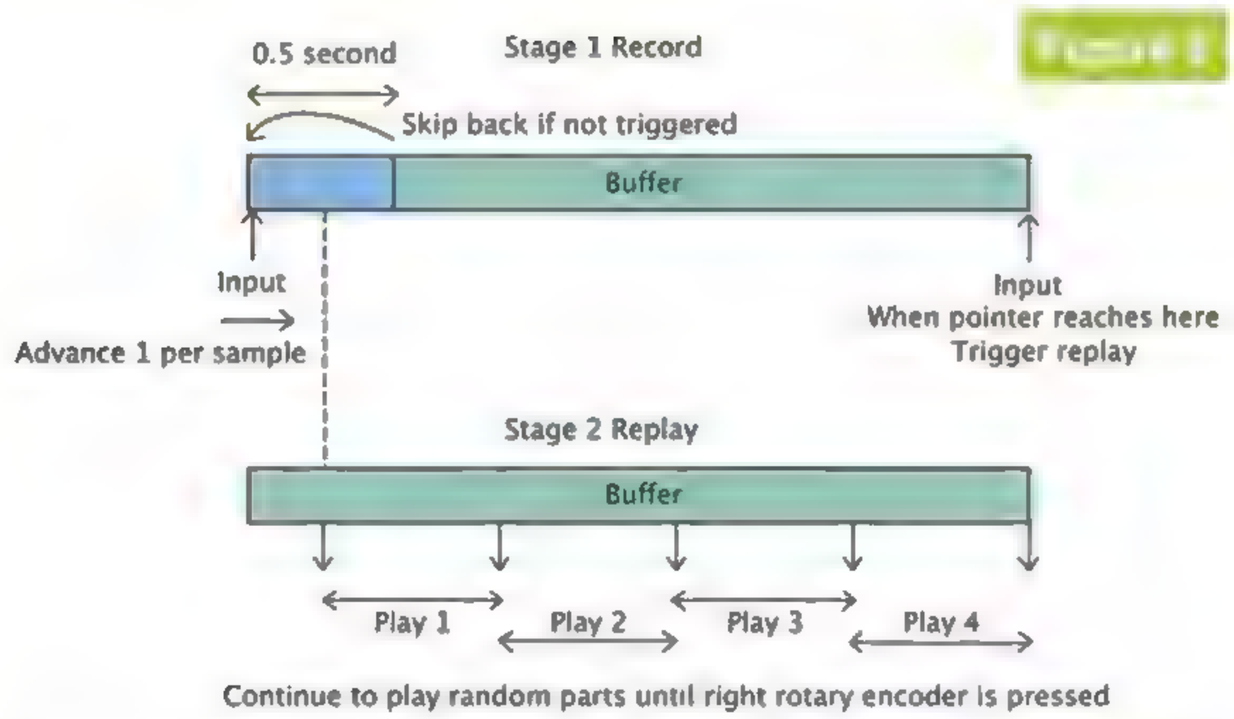
The code can be obtained from GitHub (**magpi.cc/pibakery**) and we supply both the source code and the **pico-voice.uf2** file to drag and drop onto Pico. The overall program flow is shown in **Figure 10**; it shows the program is in two parts. First off, we have a check to see if the effects number has changed – if not, which is most of the time, we call the function that creates the sample. This is done by a vector jump table for speed. Any control signal generates interrupts that either change the effects number or the parameters of the effect.

11 Finally

This set of effects are not exactly studio quality, but they show what can be done with the minimum of extra hardware. They are definitely great fun and you can spend hours exploring what sounds they can generate using different control settings. Speaking with a short delay, and learning to ignore it, has even been used as therapy for people with a stutter. Perhaps the biggest limitation on quality is with the internal ADC, which has to be said is a bit noisy. Using an external one will improve matters, as will using a DAC with more bits.

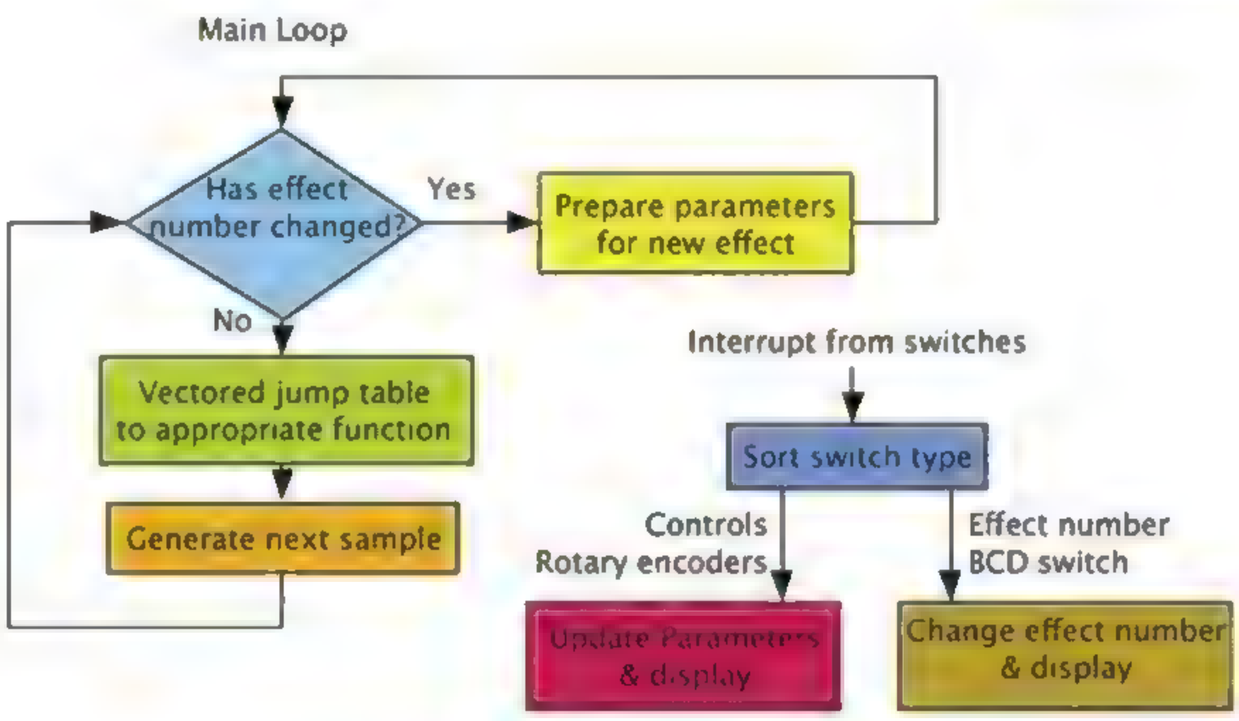


▲ **Figure 8** The vibrato effect



▲ **Figure 9** Musique concrete

▼ **Figure 10** Software program flow

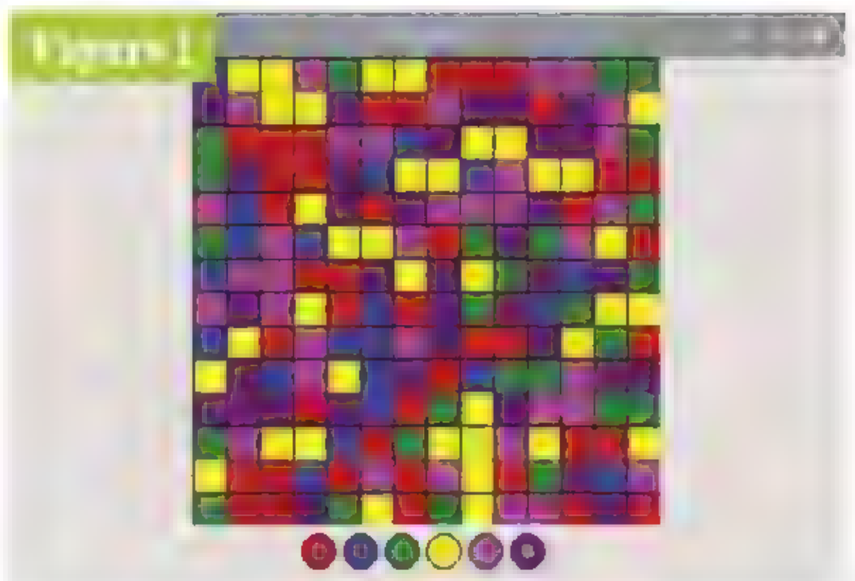


Create GUIs with Python:

Flood It

Learn how to use a Waffle to create a tasty game

Flood It is a game where the aim is to flood the board with all squares the same colour. Beginning with the top-left square, players choose a colour to flood into. It offers a slightly more complex Waffle-based game.



▶ **Figure 1** Flood the squares with one colour

Laura Sach

Laura leads the A Level team at the Raspberry Pi Foundation, creating resources for students to learn about Computer Science.

@CodeBoom

Martin O'Hanlon

Martin works in the learning team at the Raspberry Pi Foundation, where he creates online courses, projects, and learning resources.

@martinohanlon

Aim of the game

In this example (**Figure 1**), the top-left square is blue. The player could either choose to flood into the single purple square below, or to flood into the yellow square to the right.

Flooding the yellow square would be a better move because all adjoining yellow squares would also be flooded, and the player is only allowed a limited amount of moves before the game ends.

Set up the game

Download (from **magpi.cc/floodit**) and open the starter file, **floodit_starter.py**. Save it in a sensible place.

In the variables section, give the variables some values:

- `colours` – a list of six colours as strings. These can either be common colour names or hex colours. The colour names "white", "black", "red", "green", "blue", "cyan", "yellow", and "magenta" will always be available.
- `board_size` – the width/height of the board as an integer; we chose 14. The board is always a square.
- `moves_limit` – how many moves the player is allowed before they lose, as an integer; we chose 25.

In the app section, create an App widget and give it a title.

```
app = App("Flood it")
app.display()
```

Running this will result in a standard labelled window (**Figure 2**).

Create the board

The board is a grid of squares, each containing a randomly selected colour from the list you created earlier.

Inside the app, add a Waffle widget. This will create a grid which will be the board.

```
board = Waffle(app)
```

Run your program and you will see that the grid is a bit too small (**Figure 3**).

Add to the line of code you just wrote to specify parameters for the width and height of the Waffle, and make the padding between the grid squares zero.

```
board = Waffle(app, width=board_size,
height=board_size, pad=0)
```

That's better (**Figure 4**, overleaf).

Create the palette

The palette shows the player which colours they can click on to flood the board. They will click on these colours to play the game. The palette from the finished game is shown in **Figure 5**.

On the line after you created the board, create another Waffle, but this time it should be called `palette`.

Figure 2 Flood it

▲ **Figure 2** The usual labelled window

The board should start off with each square as a randomly chosen colour

```
palette = Waffle(app)
```

Remember when you added the parameters to the board Waffle in the previous step? This time, add these parameters to the palette Waffle with each one separated by a comma:

`width = 6` (the number of colours we have)
`height = 1`
`dotty = True` (this makes the squares into circles)

So, now you should have:

```
palette = Waffle(app, width=6, height=1,
dotty=True)
```

Run the code to see a blank palette (**Figure 6**).

Colour in the board

The board should start off with each square as a randomly chosen colour from the colours list you created earlier. On the line below your palette, write a call to a function

```
fill_board()
```

Find the functions section in your program, and begin writing the code for this new function:

```
def fill_board():
```

You can write a nested loop to loop through every row and column in the board. Each pixel will be coloured with a randomly chosen colour from the list. To colour in a pixel, you will use this code, where the ? symbols will be replaced with the x, y co-ordinates of the pixel:

```
board.set_pixel(?, ?, random.
choice(colours))
```

Try to write the code yourself using what you have learnt about nested loops in the previous instalments in this series – the solution is provided below if you get stuck.

Hint: *Use the board_size variable to know how many times to loop.*

```
def fill_boa
```

When you run your code, you should see a colourful board. If you see a white board, double-check that you put in the function call to `fill_board()` (**Figure 7**).

Here is one solution, but there are many ways you could do this:

```
def fill_board():
    for x in range(board_size):
        for y in range(board_size):
            board.set_pixel(x, y, random.
choice(colours))
```

An alternative solution which uses a more advanced feature called a list comprehension:

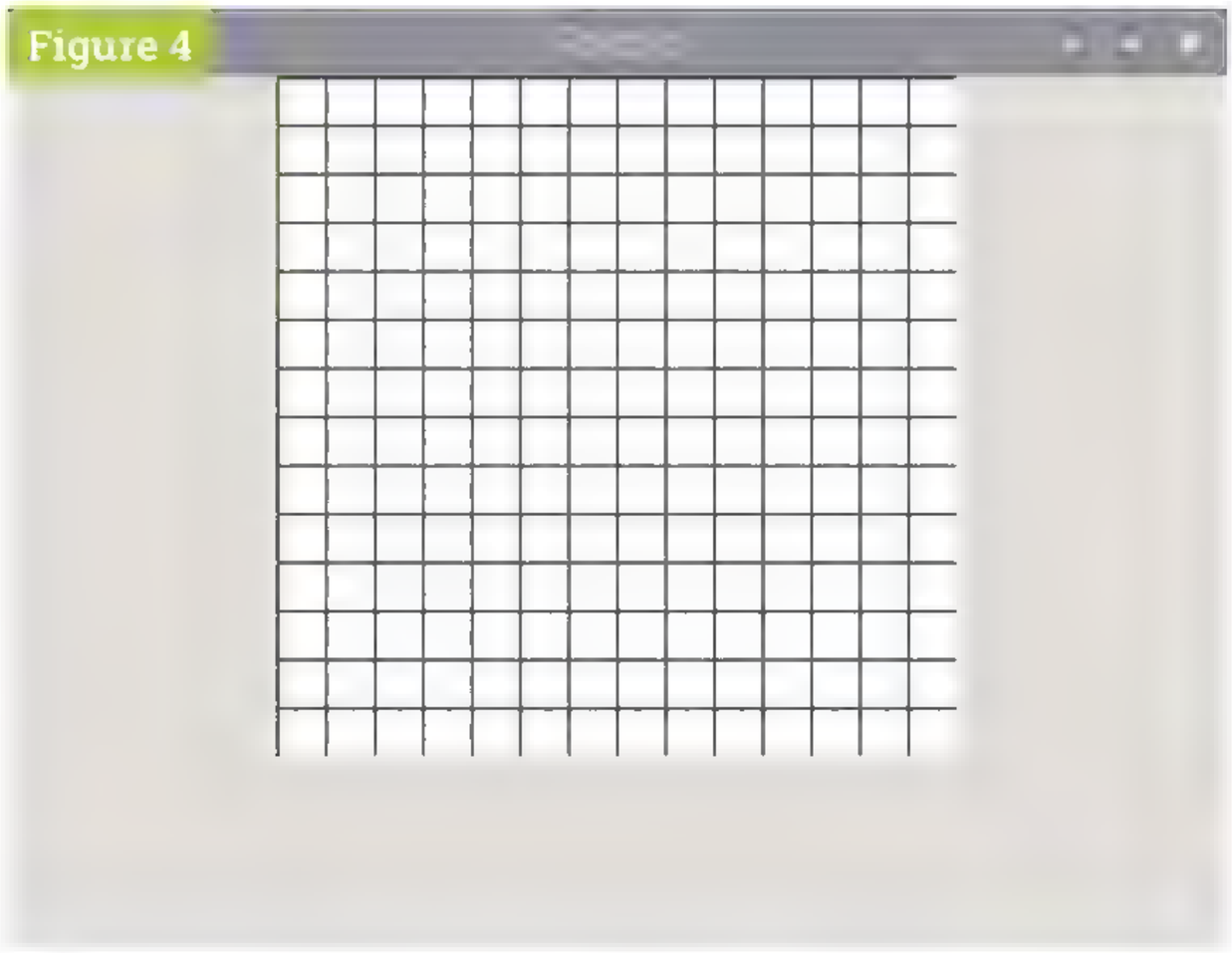
```
    board.set_pixel(x, y, random.
choice(colours))
```

Create Graphical User Interfaces with Python

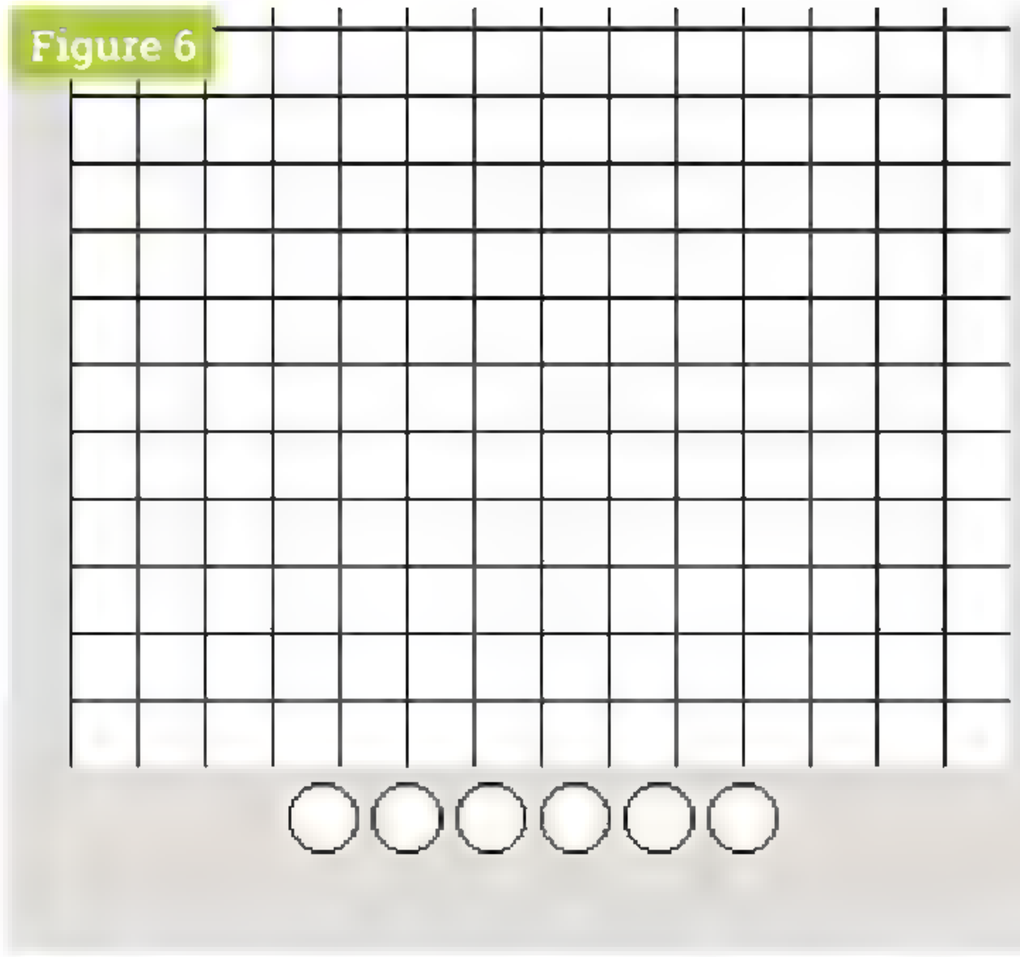
For further tutorials on how to make your own GUIs with guizero, take a look at our book, *Create Graphical User Interfaces with Python*. Its 156 pages are packed with essential info and a range of exciting projects. **magpi.cc/pythongui**

▼ **Figure 3** The grid squares are too small

Figure 3



▲ **Figure 4** A grid of the correct board size with no padding



▶ **Figure 5** A blank palette

Colour in the palette

Now that you have a colourful board, let's colour in the palette. On the line below your `fill_board()` code, write a call to a function:

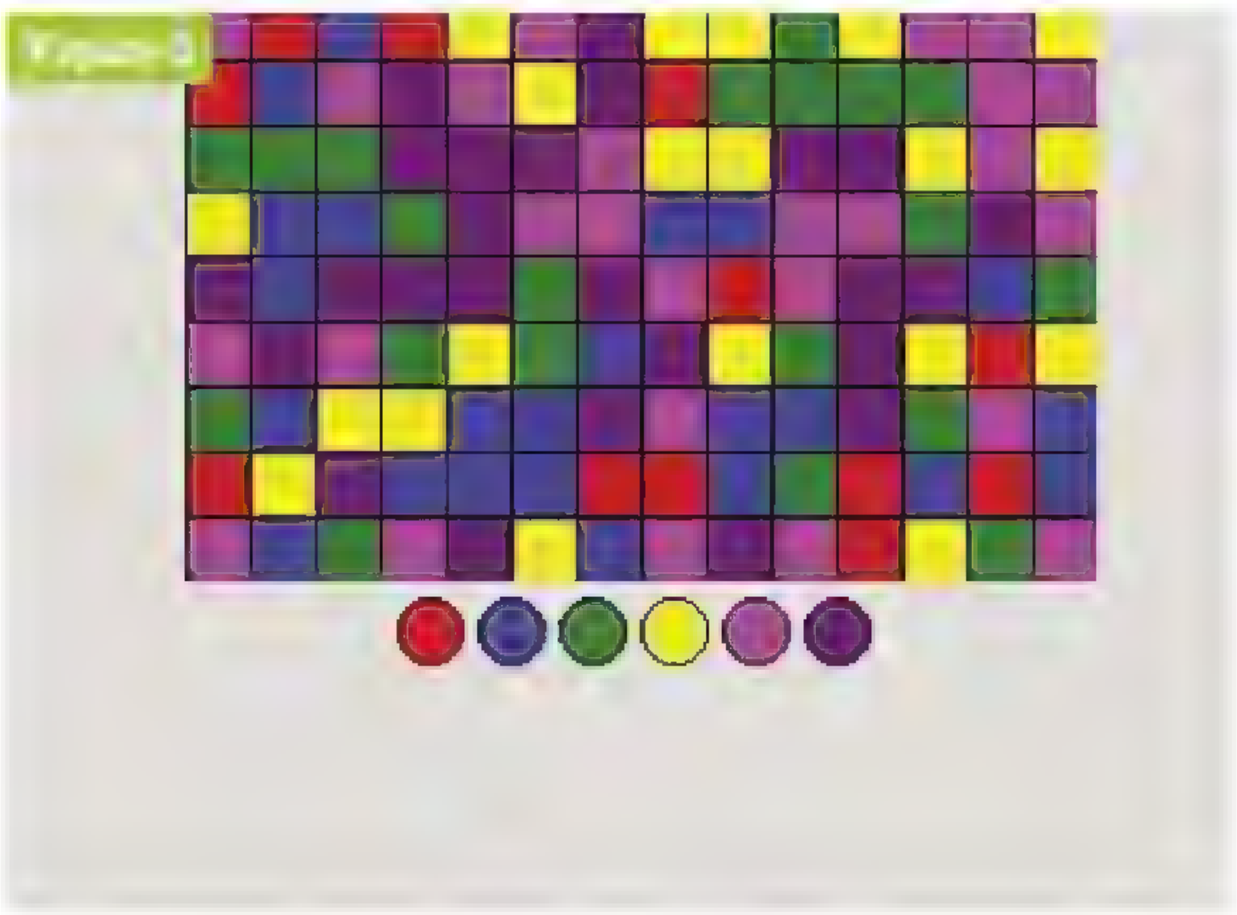
```
init_palette()
```

Find the functions section in your program, and begin writing the code for this new function:

```
def init_palette():
```

The idea here is to loop through all of the colours in the list, assigning one to each of the circles in the palette. You can use the same `set_pixel` method as you used for the board to change the colour of the circles in the palette.

▼ **Figure 5** You'll need a palette for the player to choose a colour



Have a go at writing the code yourself. If you get stuck, see the 'How many ways can you colour the palette' box overleaf for some possible solutions.

***Hint**: All of the circles in the palette are in row 0 of the Waffle.*

Start the flood

When the player clicks on a colour in the palette, the board should flood with that colour, beginning with the top-left square.

In the functions section, create a new function called `start_flood` in exactly the same way as you did for the last two functions. This function needs to take two parameters which will be the x, y co-ordinates of the square that was clicked on. Add these between the brackets so that you end up with your code looking like this:

```
def start_flood(x, y):
```

Add a line of code (indented) to the function to obtain the name of the colour that was clicked on:

```
flood_colour = palette.get_pixel(x,y)
```

This will be the colour to flood the board with

Now add a line of code to get the current colour of the starting pixel – this is always the pixel in the top left of the board, at co-ordinates 0, 0.

```
target = board.get_pixel(0,0)
```

Now call the `flood` function, which has already been written for you in the starter file. This function starts at 0,0 and floods all the pixels

connected to the top-left pixel that are the same colour with the `flood_colour`.

```
flood(0, 0, target, flood_colour)
```

This function should run whenever someone clicks on a colour in the palette, so find the line of code where you created the palette.

```
palette = Waffle(app, width=6, height=1,
dotty=True)
```

Add another parameter, which is a command. When a circle on the palette is clicked, this command will be executed. The command is the function `start_flood`, so your code should now look like this:

```
palette = Waffle(app, width=6, height=1,
dotty=True, command=start_flood)
```

Test out your code by clicking on the circles on the palette.

In our random example (**Figure 8**), the top-left square is green. If you click purple on the palette, the top-left square will turn purple and connect to the purple square below (**Figure 9**, overleaf).

Now there are five purple squares connected to the top-left square. Let's click pink to connect up the pink squares underneath (**Figure 10**).

Now there is a large chain of pink squares. Continue the game by pressing different colours in the palette. The aim is to eventually get all of the squares the same colour.

Figure 7

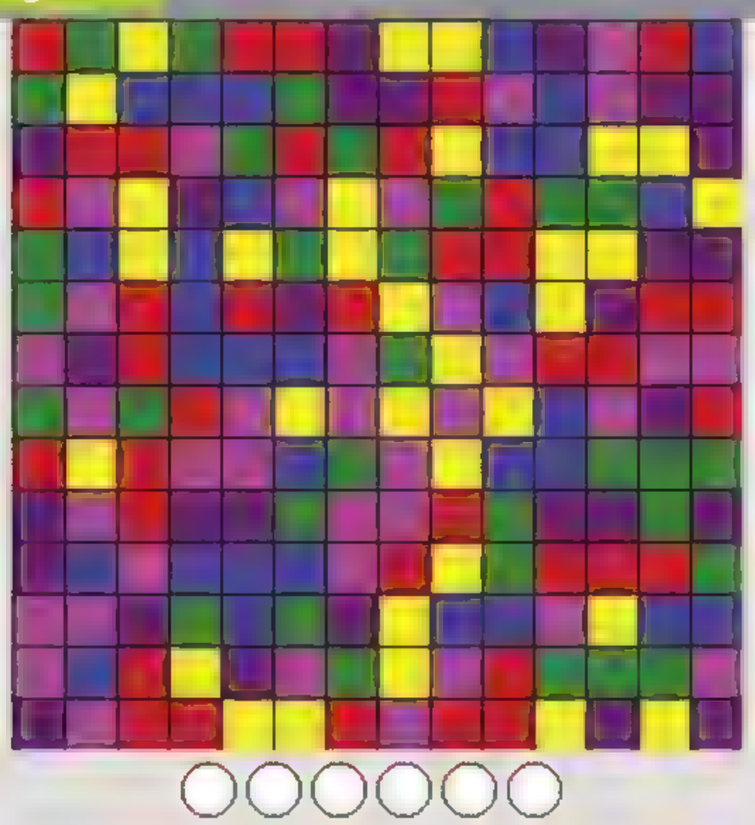


Figure 7 Each square of the board is coloured randomly

Figure 8

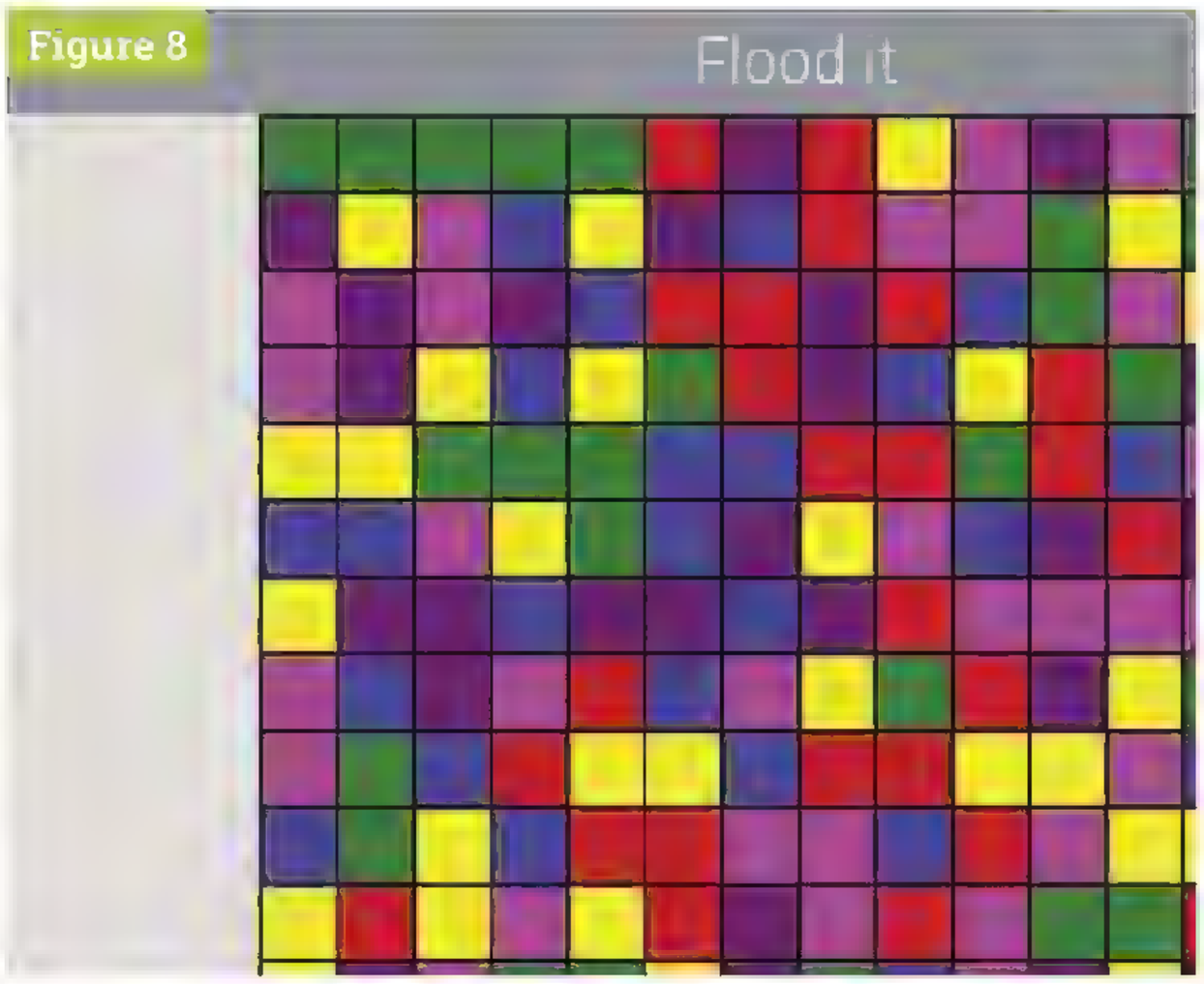


Figure 8 Here, the top-left square is green

> Add a piece of text to the GUI to display whether the player has won or lost

Winning the game

At the moment, if the player manages to get all of the squares in the grid the same colour, nothing happens. The player is also allowed an infinite number of turns, as the number of moves they have taken is not tracked.

First let's add a piece of text to the GUI to display whether the player has won or lost. The text will start off blank.

Underneath the code for the palette, add a Text widget called `win_text`.

```
win_text = Text(app)
```

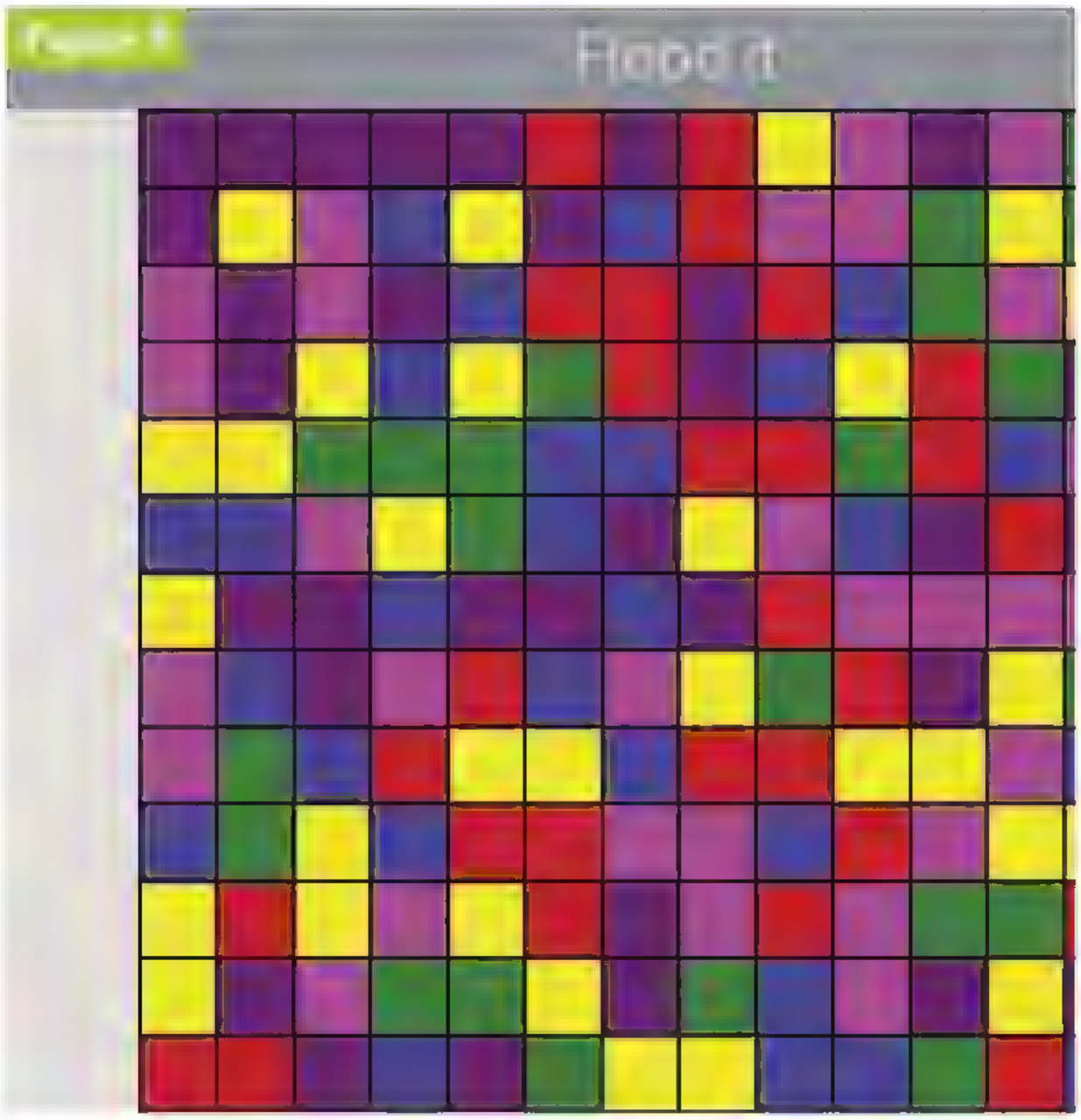
In the variables section, add another variable called `moves_taken` and set it to 0.

Now create a function called `win_check`, which will check after each move whether the player has won.

First, you need to specify that you would like to be allowed to change the value of the global variable `moves_taken`.

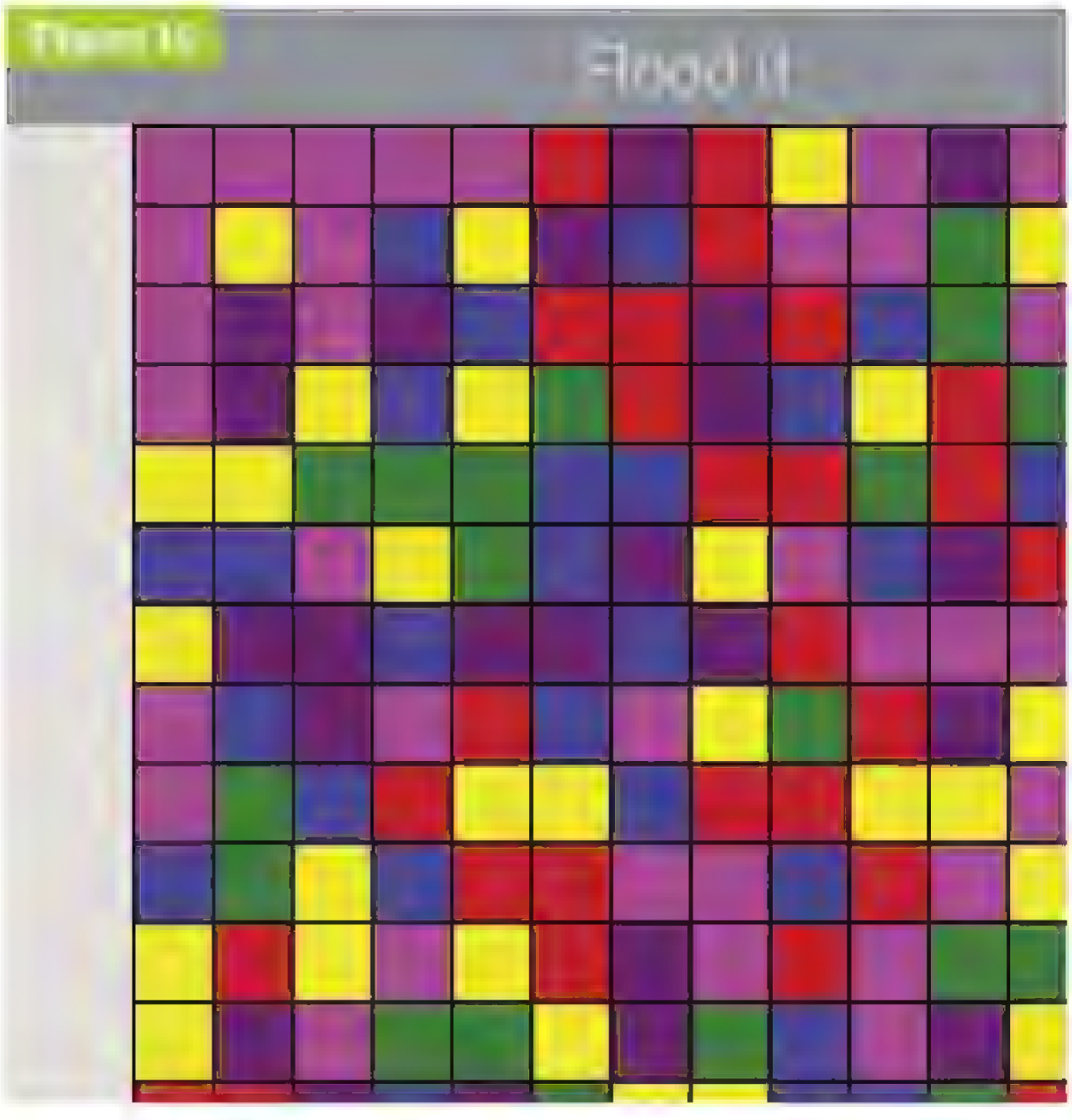
```
global moves_taken
```

Then add 1 to the `moves_taken` variable – each time this function is called, we will add one more move.



▲ **Figure 9** Clicking purple turns it purple

▼ **Figure 10** Click pink for a chain of pink




```
moves_taken += 1
```

Check whether the `moves_taken` is less than the `moves_limit` or not:

```
if moves_taken < moves_limit:

else:
```

If the `moves_taken` is not within the limit, this means the playher has run out of moves, so update the text to say that they lost:

```
if moves_taken < moves_limit:

else:
    win_text.value = "You lost :("
```

If the number of moves taken is less than the limit, check whether all of the squares are the same colour by calling the function already written for you in the starter file. Make sure the following code is indented below the first `if` statement:

```
if all_squares_are_the_same():
    win_text.value = "You win!"
```

The completed piece of code should look like this:

```
def win_check():
    moves_taken += 1
    if moves_taken <= moves_limit:
        if all_squares_are_the_same():
            win_text.value = "You win!"
    else:
        win_text.value = "You lost :("
```

Finally, you must call the `win_check` function whenever a square is clicked on. The easiest way to do this is to add the function call at the end of the `start_flood` function.

Now it's time to test the game. An example code listing is shown in **08-floodit.py**.

Test your game

You can test whether the game works by playing it; however, it might take a long time to test whether you can win! An easier way to check is to change the `board_size` variable to something small such as 5, and then play the game on a much smaller grid to see whether you can win.

You can easily test whether the game causes you to lose properly by clicking on the same colour 25 times!

How many ways can you colour the palette?

Here is a solution which uses a loop and a variable to keep track of which column you are colouring in:

```
def init_palette():
    column = 0
    for colour in colours:
        palette.set_pixel(column, 0, colour)
        column += 1
```

Here is a similar solution which uses a **range** inside the **for** loop instead of a counter variable:

```
def init_palette():
    for x in range(len(colours)):
        palette.set_pixel(x, 0, colours[x])
```

Here is a different solution which uses the index function **colours.index(colour)**. This code says 'In the colours list, find me the position in the list of colour'. So, for example if your list was **["green", "blue", "red"]** then the index of green would be **0**, the index of blue would be **1**, etc., remembering that we start counting from zero.

```
def init_palette():
    for colour in colours:
        palette.set_pixel(colours.index(colour), 0, colour)
```

You can use any of these solutions, or you may have come up with a different way by yourself. None of them is the 'right answer': there are often many different ways of coding a solution.

08-floodit.py

Language: **Python 3**

DOWNLOAD THE FULL CODE:
magpi.cc/floodit

```
# ----------------------------
# Imports
# ----------------------------

from guizero import App, Waffle, Text, PushButton, info
import random

# ----------------------------
# Variables
# ----------------------------

colours = ["red", "blue", "green", "yellow", "magenta",
"purple"]
board_size = 14
moves_limit = 25
moves_taken = 0

# ----------------------------
# Functions
# ----------------------------

# Recursively floods adjacent squares
def flood(x, y, target, replacement):
    # Algorithm from https://en.wikipedia.org/wiki/
Flood_fill
    if target == replacement:
        return False
    if board.get_pixel(x, y) != target:
        return False
    board.set_pixel(x, y, replacement)
    if y+1 <= board_size-1:   # South
        flood(x, y+1, target, replacement)
    if y-1 >= 0:              # North
        flood(x, y-1, target, replacement)
    if x+1 <= board_size-1:    # East
        flood(x+1, y, target, replacement)
    if x-1 >= 0:              # West
        flood(x-1, y, target, replacement)

# Check whether all squares are the same
def all_squares_are_the_same():
    squares = board.get_all()
    if all(colour == squares[0] for colour in squares):
        return True
    else:
        return False

def win_check():
    global moves_taken
    moves_taken += 1
    if moves_taken <= moves_limit:
        if all_squares_are_the_same():
            win_text.value = "You win!"
    else:
        win_text.value = "You lost :("

def fill_board():
    for x in range(board_size):
        for y in range(board_size):
            board.set_pixel(x, y, random.choice(colours))

def init_palette():
    for colour in colours:
        palette.set_pixel(colours.index(colour), 0,
colour)

def start_flood(x, y):
    flood_colour = palette.get_pixel(x,y)
    target = board.get_pixel(0,0)
    flood(0, 0, target, flood_colour)
    win_check()

# ----------------------------
# App
# ----------------------------

app = App("Flood it")

board = Waffle(app, width=board_size,
height=board_size, pad=0)
palette = Waffle(app, width=6, height=1, dotty=True,
command=start_flood)

win_text = Text(app)

fill_board()
init_palette()

app.display()
```

Get to grips with administering Pi-hole

Explore the Pi-hole dashboard, learn how to beef up its web filtering features, and use its integrated DHCP server

Nik Rawlinson

Esperanto-speaking, pencil-wielding, single-board computing fan who likes hyphens and remembers what that icon on the save button depicts.

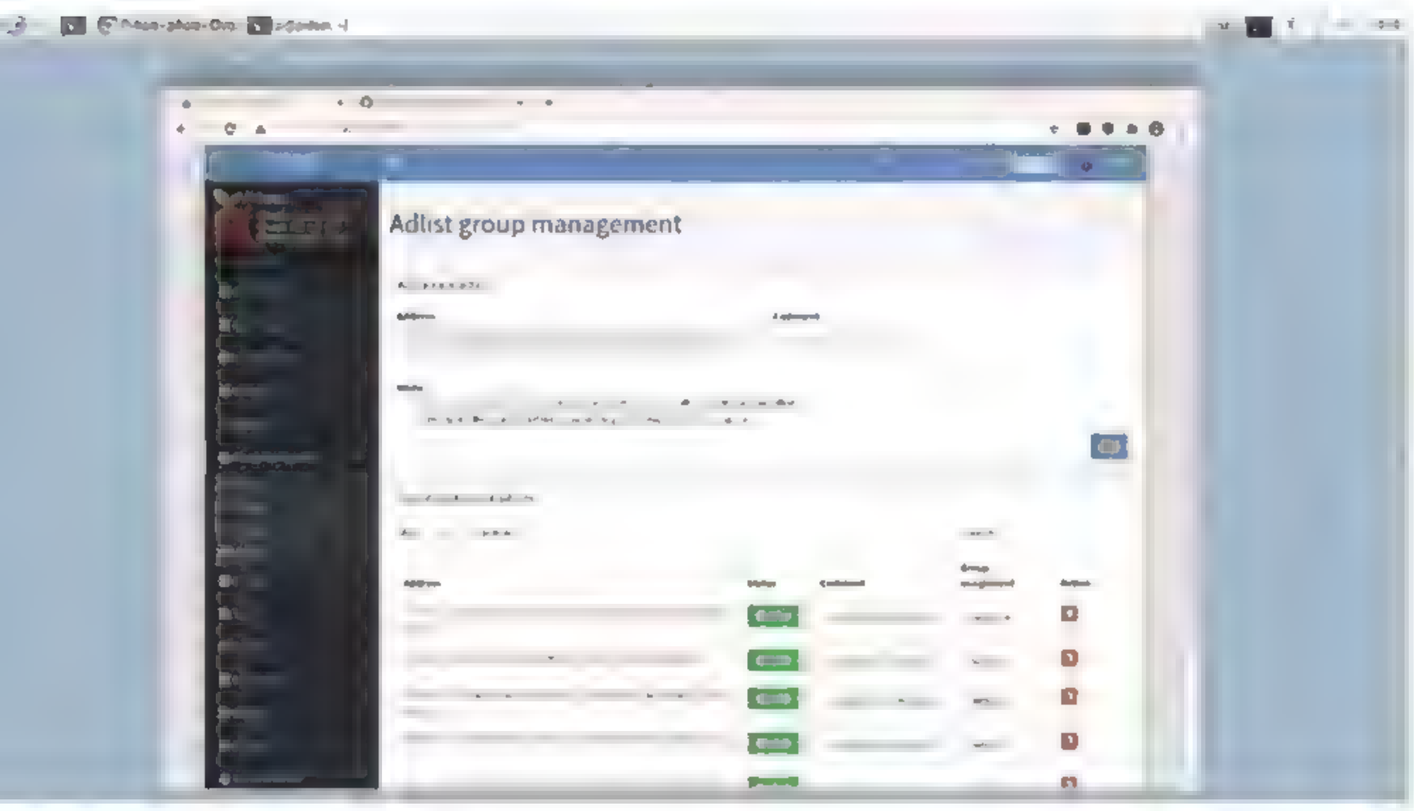
nikrawlinson.com

In the last two workshops, we showed you how to set up Pi-hole to filter ads and other web content on your home network, and how to change the DNS settings so you have more granular control over the way it works. In this final instalment, we'll be taking a look at the broader Pi-hole dashboard, and showing you how the changes you make there will allow you to control what is – and isn't – available to other computers on your network. We'll integrate third-party domain lists to block distractions with less effort, and use Pi-hole's integrated DHCP server to simplify setup on connected clients.

Using third-party domain lists simplifies the task of adding comprehensive sets of addresses for social networking and other services

01 Log in to Pi-hole

Although Pi-hole runs on Raspberry Pi, you can access the dashboard by pointing a browser towards **pi.hole/admin/** on any computer that uses Pi-hole to filter its web content. Other devices on your network, not filtered by Pi-hole, can also access the dashboard by adding '/admin/' to the end of the numeric IP address of the Pi-hole device. Click Login in the sidebar and provide your password. Once you've logged in, you'll notice that the dashboard suddenly becomes a lot more informative. It's worth spending some time getting familiar with its contents.

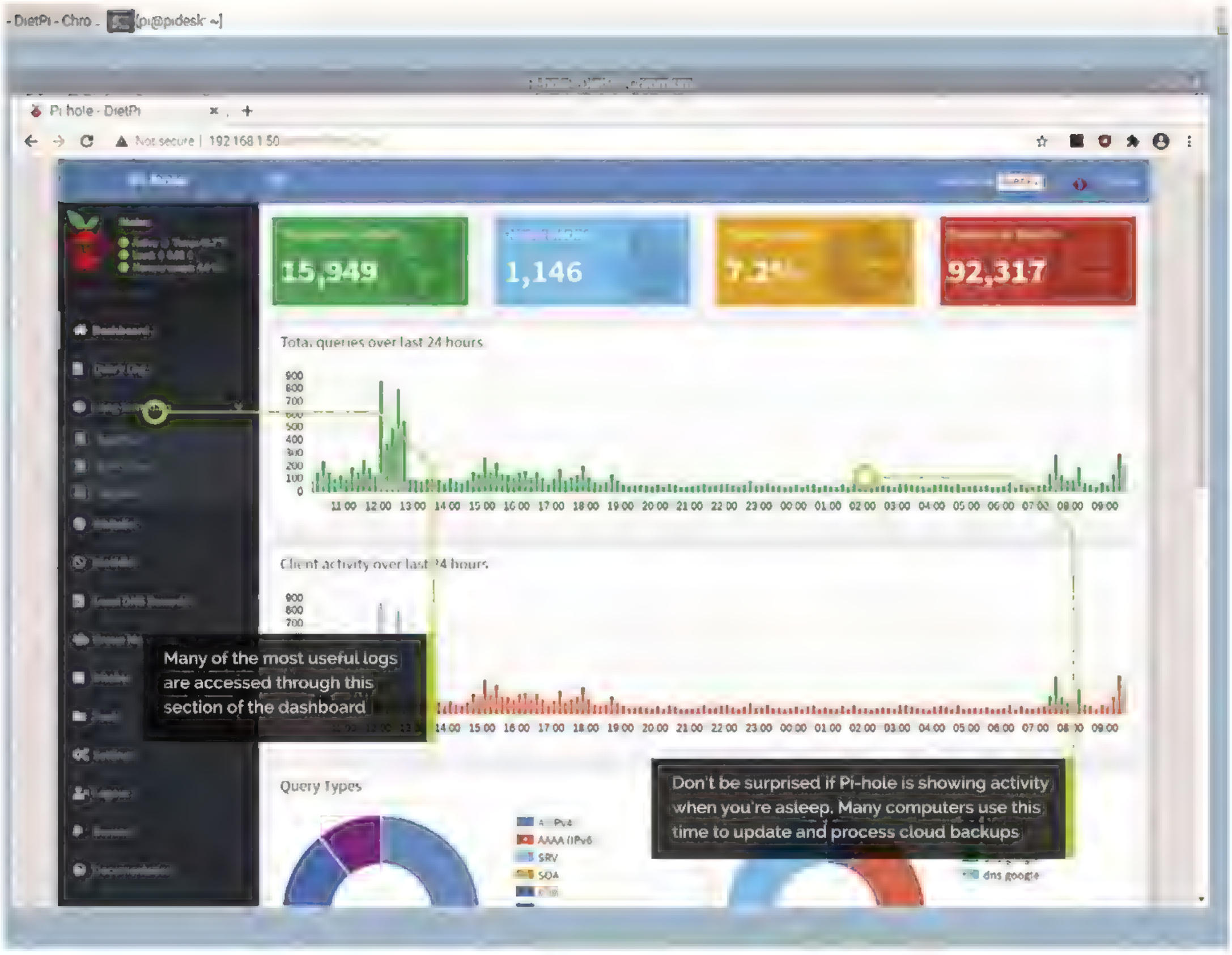


02 Understand your traffic

The uppermost two graphs depict DNS queries passing through Pi-hole. Don't be surprised if there are overnight queries, when backup services kick in and some computers may be set to update. You can see which of your devices is making the most calls at the bottom of the page, where their activity is split out into total requests and blocked requests. The 'Queries answered by' panel shows you a count of where the answer for each query was sourced from.

03 Check what's being blocked

Unless you disabled logging when you installed Pi-hole, it will make a note of every request it processes, whether permitted or blocked. Browsing the log is a good way to check whether anything on your network is attempting to make persistent connections to an unauthorised remote server, which could suggest it's hosting malware. Click 'Long-term data' and, in the section this exposes, click Top Lists. Choose a time frame by



clicking 'Click to select date and time range'. Start with 'Last 7 days' and browse the entries in the Top Blocked Domains box. If anything looks suspicious, check it using a service like Kaspersky's Threat Intelligence Portal (click Lookup at **opentip.kaspersky.com**).

04 Investigate frequent callers

Should you believe the domain in question is a threat, investigate which computer – or computers – on your network is making the requests to that address. Click Query Log, immediately above Top Lists, and isolate the same time frame by once again clicking 'Click to select date and time range'. Now type the domain in question into the Search box and check the IP addresses in the Client column to see which computers on your network are insistently reaching out to it. Scan the affected computers for threats using your regular anti-malware tools.

05 Blacklist suspicious domains

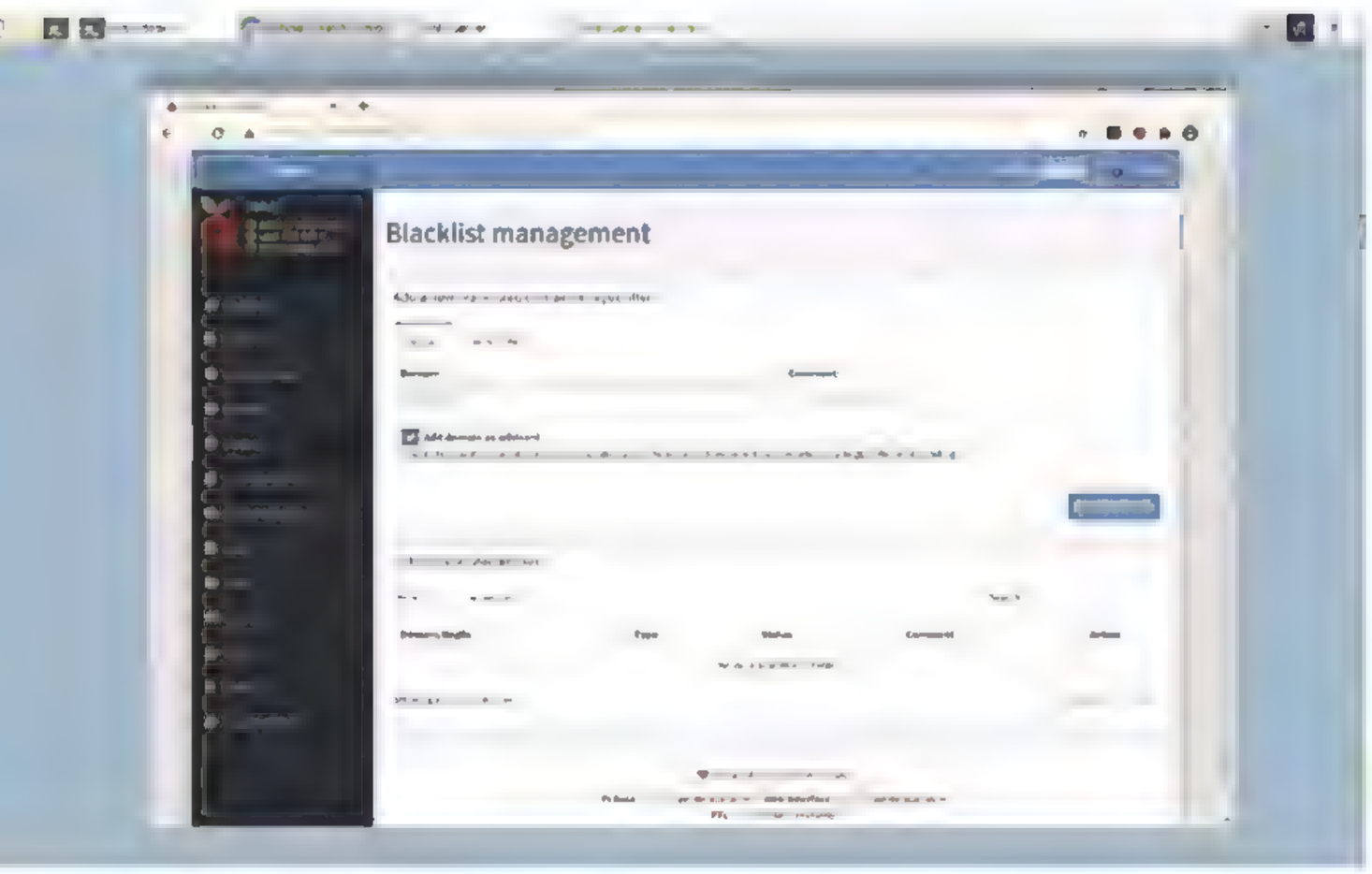
If you're sure the domain is a threat, you can also block access to it through Pi-hole, which will stop any malware running on computers filtered by your Pi-hole setup from reaching it, potentially preventing them from stealing your personal data. Simply click Blacklist in the Action column against the domain you want to bar. Doing so will only affect the specific domain and subdomain as specified in the Domain column, but leave other subdomains active, so this isn't the best way to block, say, social networking sites, which frequently use different domains for browser and app access.

06 Block complete domains

So, if you want to use Pi-hole to block access to complete services, click Blacklist in the sidebar and type the last part of the domain name into the Domain box. For example, **bbc.co.uk**, but

You'll Need

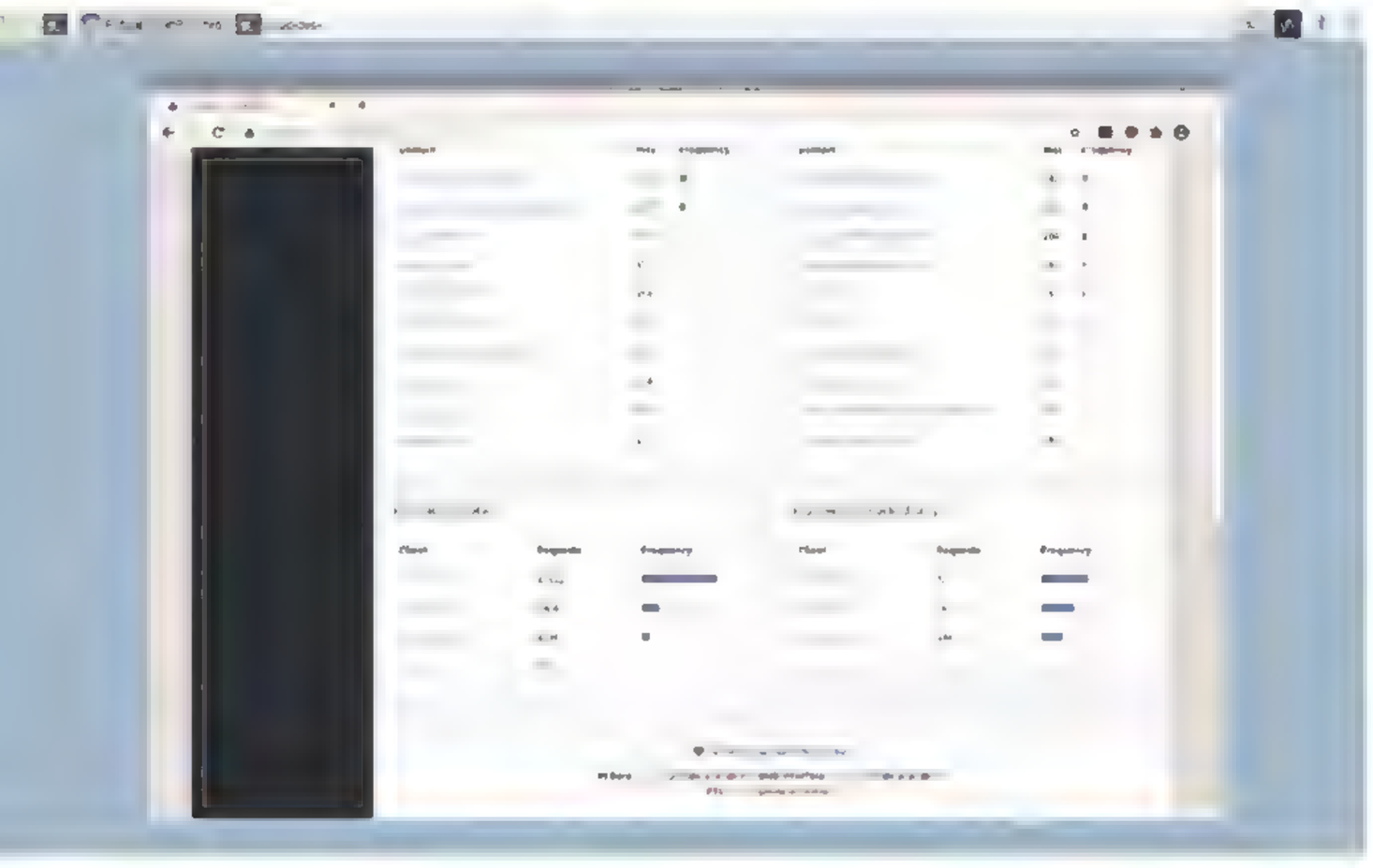
- Raspberry Pi
- Pi-hole **pi-hole.net**
- Web browser



▲ Adding a domain to the blacklist or whitelist and ticking the wildcard box will apply the same rule to all of its subdomains

▶ Use the timeout feature to temporarily disable Pi-hole. This is useful when diagnosing connectivity issues on client devices, or for one-off visits to blocked sites

▼ Check the status box at the bottom of the dashboard for updates to the Pi-hole software, and patch it as required



not **www.bbc.co.uk** if you want to block the BBC. Now, to block all subdomains within bbc.co.uk, including those hosting the website, feeding data to apps, serving video and so on, click 'Add domain as wildcard', then click 'Add to Blacklist'. In much the same way, if any services you need to use are being blocked by the third-party filter lists Pi-hole uses, you can specifically allow them by clicking Whitelist in the sidebar and adding them there.

07 Bolster your social blocks

Note that some services use multiple domains, which will all need blacklisting if you want to lock them out entirely. For example, Facebook also hosts data at facebook.net, fbcdn.com, fbcdn.net, fbsbx.com, and more. Rather than researching every possible domain for each social network, you can incorporate third-party lists, like those maintained on GitHub at **magpi.cc/socialblocklists**. In Pi-hole, click Group Management in the sidebar, followed by Adlists. Now paste each pihole-prefixed list into the Address box in turn (for example, **https://raw.githubusercontent.com/gieljnssns/Block_facebook_dns/master/pihole-facebook.txt**), clicking Add after each one.

08 Take a timeout

If you briefly need to visit a site that's blocked by Pi-hole, or you're trying to diagnose a problem that you suspect the filter might be causing, you can temporarily disengage the whole system, rather than trying to work out which domains you need to unlock. Click Disable in the sidebar, then choose how long you'd like to switch off for. The default options are 10 seconds, 30 seconds, and five minutes, but you can also pause filtering indefinitely, or click 'Custom time' and enter your preferred duration. This exposes every device previously filtered using Pi-hole, not only the one through which you're accessing the dashboard.

09 Assign network addresses

So far, we've assumed you're going to continue using your router's DHCP server to assign numeric addresses to clients on your network, which we'd recommend if you want to use Pi-hole on just a few of your devices. However, if you intend

to use it to filter content for every machine you own, it makes sense to also enable its DHCP server. Doing so means they'll automatically use Pi-hole as their DNS server, too, which simplifies configuration and means any new computers – and devices like voice assistants and smart TVs – will be secured as soon as they connect to your network.

10 DHCP server

Click Settings, then click the DHCP tab. Click the checkbox beside 'DHCP server enabled', then use the From and To boxes to specify the range of addresses that Pi-hole can hand out. Unless you have a lot of devices on your network, you should be safe to leave the defaults as they stand, giving you capacity for 50 devices. Check that the address of your router, as specified in the Router step below, is correct. If you don't know what your router's address is, open a Terminal window and type:

```
ip route | grep default
```

The router address appears immediately after 'via'.

11 Disable router-based DHCP

You shouldn't have two devices handing out IP addresses on your network, so log in to your router's admin pages through the browser

> You shouldn't have two devices handing out IP addresses on your network

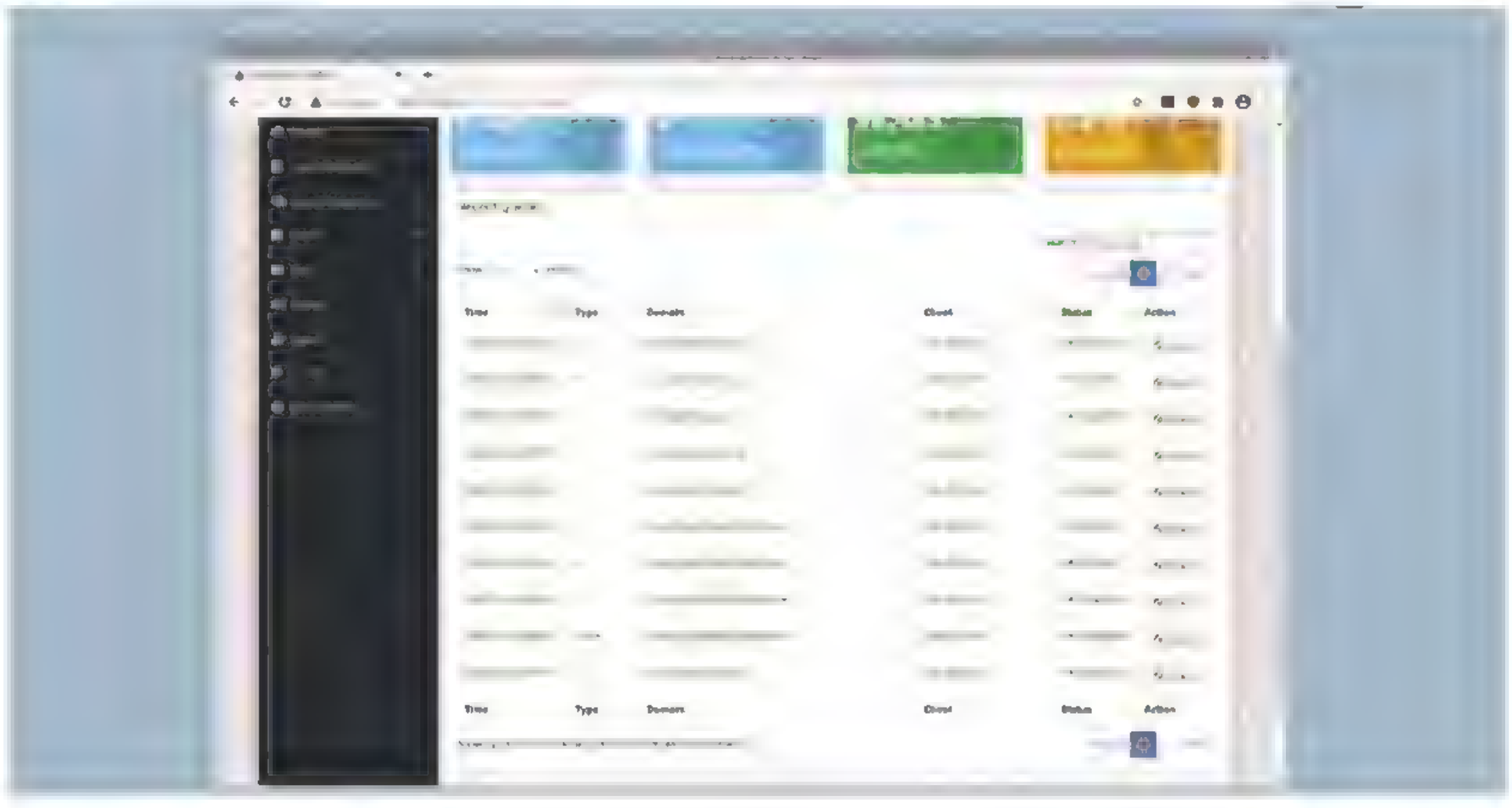
by typing in the address you obtained using the `ip route` command in the previous step. Navigate through its various screens until you find the DHCP setting. If this is protected, the administrator password is often specified on the back of the router itself. Clients on your network will seek out a replacement DHCP server automatically and, with your router no longer providing this function, should find your Pi-hole installation, which will assign them the address they require.

12 Keep Pi-hole updated

The status bar at the bottom of every dashboard page will warn you if your installation is out of date. At the moment, it's not possible to update the system via the graphical interface. So, if you spot a red, pulsing 'Update available!' beside the Pi-hole, web interface or FTL version numbers, open a Terminal window on the Raspberry Pi running Pi-hole and enter:

```
pihole -up
```

Pi-hole will check for the latest available version, then download and install it automatically.



When you've queried your logs, you can add domains to your blacklist directly by clicking the appropriate button

Top Tip

Terminal tweaks

TPi-hole can be tweaked and updated at the Terminal, too. Check **magpi.cc/piholecommand** for a comprehensive list of commands.

Top Tip

Double servings

Run two Pi-hole servers on your network and if one falls over, devices can switch to the other until it returns.

Raspberry Pi Pico temperature gauge

Use your Raspberry Pi Pico's built-in ADC to convert analogue inputs, and read its internal temperature sensor

Gareth Halfacree

With a passion for open-source software and hardware, Gareth was an early adopter of the Raspberry Pi platform and has written several publications on its capabilities and flexibility.

@ghalfacree

In previous tutorials you've been using the digital inputs on your Raspberry Pi Pico. A digital input is either on or off, a binary state. When a push-button switch is pressed, it changes a pin from low, off, to high, on; when a passive infrared sensor detects motion, it does the same.

Your Pico can accept another type of input signal, though: analogue input. Whereas digital is only ever either on or off, an analogue signal can be anything from completely off to completely on – a range of possible values. Analogue inputs are used for everything from volume controls to gas, humidity, and temperature sensors – and they work through a piece of hardware known as an analogue-to-digital converter (ADC).

In this guide, you'll learn how to use the ADC on your Pico – and how to tap in to its internal temperature sensor to build a data-logging heat measurement gadget. You'll also learn a technique for creating an analogue-like output. For this you'll need your Pico; an LED of any colour and 330 Ω resistor; a 10 kΩ potentiometer; and a selection of male-to-male (M2M) jumper wires. You'll also need a micro USB cable, and to connect your Pico to your Raspberry Pi or other computer running the Thonny MicroPython IDE.

You'll Need

- Raspberry Pi Pico
- Breadboard
- LED (any colour)
- 330 Ω resistor
- 10 kΩ potentiometer
- Male-to-male (M2M) jumper wires

The analogue-to-digital converter

Raspberry Pi Pico's RP2040 microcontroller is a digital device, like all mainstream microcontrollers: it is built up of thousands of transistors, tiny switch-like devices which are either on or off. As a result, there's no way for your Pico to truly understand an analogue signal – one which can be anything on a spectrum between fully off and fully on – without relying on an additional piece of hardware: the analogue-to-digital converter (ADC).

As the name suggests, an analogue-to-digital converter takes an analogue signal and changes it to a digital one. You won't see the ADC on your Pico, no matter how closely you look: it's built into RP2040 itself. Many microcontrollers have their own ADCs, just like RP2040, and the ones that don't can use an external ADC connected to one or more of their digital inputs.

An ADC has two key features: its resolution, measured in digital bits, and its channels, or how many analogue signals it can accept and convert at once. The ADC in your Pico has a resolution of 12 bits, meaning that it can transform an analogue signal into a digital signal as a number ranging from 0 to 4095 – though this is handled in MicroPython transformed to a 16-bit number ranging from 0 to 65,535, so that it behaves the same as the ADC on other MicroPython microcontrollers. It has three channels brought out to the GPIO pins: GP26, GP27, and GP28, which are also known as GP26_ADC0, GP27_ADC1, and GP28_ADC2 for analogue channels 0, 1, and 2. There's also a fourth ADC channel, which is connected to a temperature sensor built into RP2040; you'll find out more about that later in the tutorial.

> An ADC takes an analogue signal and changes it to a digital one

Reading a potentiometer

Every pin connected to your Pico's analogue-to-digital converter can also be used as a simple digital input or output; to use it as an analogue input, you'll need an analogue signal – and you can easily make one with a potentiometer.

There are various potentiometer types available: some, like the ones on the HC-SR501 passive IR sensor we used last issue (**magpi.cc/106**), are designed to be adjusted with a screwdriver; others, often used for volume controls and other inputs, have knobs or sliders. The most common type has a small, usually plastic, knob coming out of the top or front: this is known as a rotary potentiometer.

Pick up your potentiometer and turn it over: you'll see it has three pins which fit in the breadboard. Depending on how you wire these pins up, the potentiometer works in two different ways. Start by inserting the potentiometer into your breadboard, being careful not to bend the pins. Wire the middle pin to pin GP26_ADC0 on your Pico using a male-to-male (M2M) jumper wire, as shown in **Figure 1** – if your Pico is inserted into the breadboard at the very top, it'll be on row 10. Finally, take two more jumper wires and wire one of the potentiometer's outer pins – it doesn't matter which – to your breadboard's power rail and the power rail to your Pico's 3V3 pin.

Open Thonny and begin a new program:

```
import machine
import utime
```

Like the digital general-purpose input/output (GPIO) pins, the analogue input pins are handled by the machine library – and just like the digital pins, they need to be set up before you can use them. Continue your program:

```
potentiometer = machine.ADC(26)
```

This configures pin GP26_ADC0 as the first channel, ADC0, on the analogue-to-digital converter. To read from the pin, set up a loop:

```
while True:
    print(potentiometer.read_u16())
    utime.sleep(2)
```

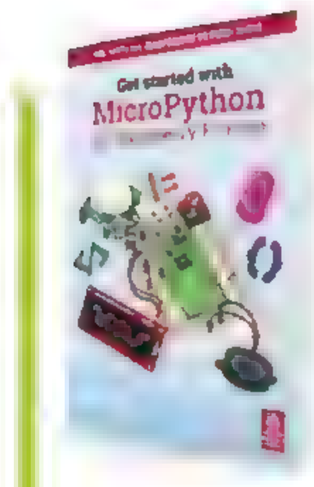
In this loop, reading the value of the pin and printing it take place on a single line: this is a more compact alternative to reading the value into a variable and then printing the variable, but only works if you don't want to do anything with the reading other than print it – which is exactly what this program needs at the moment.

Reading an analogue input is almost identical to reading a digital input, except for one thing: when reading a digital input you use `read()`, but this analogue input is read with `read_u16()`. That last part, `u16`, simply warns you that rather than receiving a binary 0 or 1 result, you'll receive an unsigned 16-bit integer – a whole number between 0 and 65,535.

Click the Run icon and save your program as **Potentiometer.py**. Watch the Shell: you'll see your program print out a large number, likely over 60,000. Try turning the potentiometer all the way in one direction: depending on the direction you turned the knob and the outer leg you used in your circuit, the number will go up or down. Turn it the other way: the value will change in the opposite direction.

No matter which way you turn it, though, it will never get anywhere near 0. That's because with only two legs connected, the potentiometer is acting as a component known as a variable resistor or varistor. A varistor is a resistor with a value you can change – in the case of a 10 kΩ potentiometer, between 0 Ω and 10,000 Ω. The higher the resistance, the less voltage from the 3V3 pin reaches your analogue input – so the number goes down. The lower the resistance, the more voltage reaches your analogue input – so the number goes up.

A potentiometer works by having a conductive strip inside, connected to the two outer pins, and a wiper or brush connected to the inner pin (**Figure 2**, overleaf). As you turn the knob, the wiper moves closer to one end of the strip and further away from the other. The further the wiper gets from the end of the strip you wired to your Pico's 3V3 pin, the higher the resistance; the closer it gets, the lower the resistance.



Get Started with MicroPython on Raspberry Pi Pico

For more physical computing projects to try on your Raspberry Pi Pico, grab a copy of the new book, *Get Started with MicroPython on Raspberry Pi Pico*. As well as learning how to use Raspberry Pi Pico's pins as inputs and outputs, you'll build a simple game, measure temperatures, save and load data to your Pico's file system, and even make a burglar alarm for your room. *Get Started with MicroPython on Raspberry Pi Pico* is available now from **magpi.cc/picobook**.

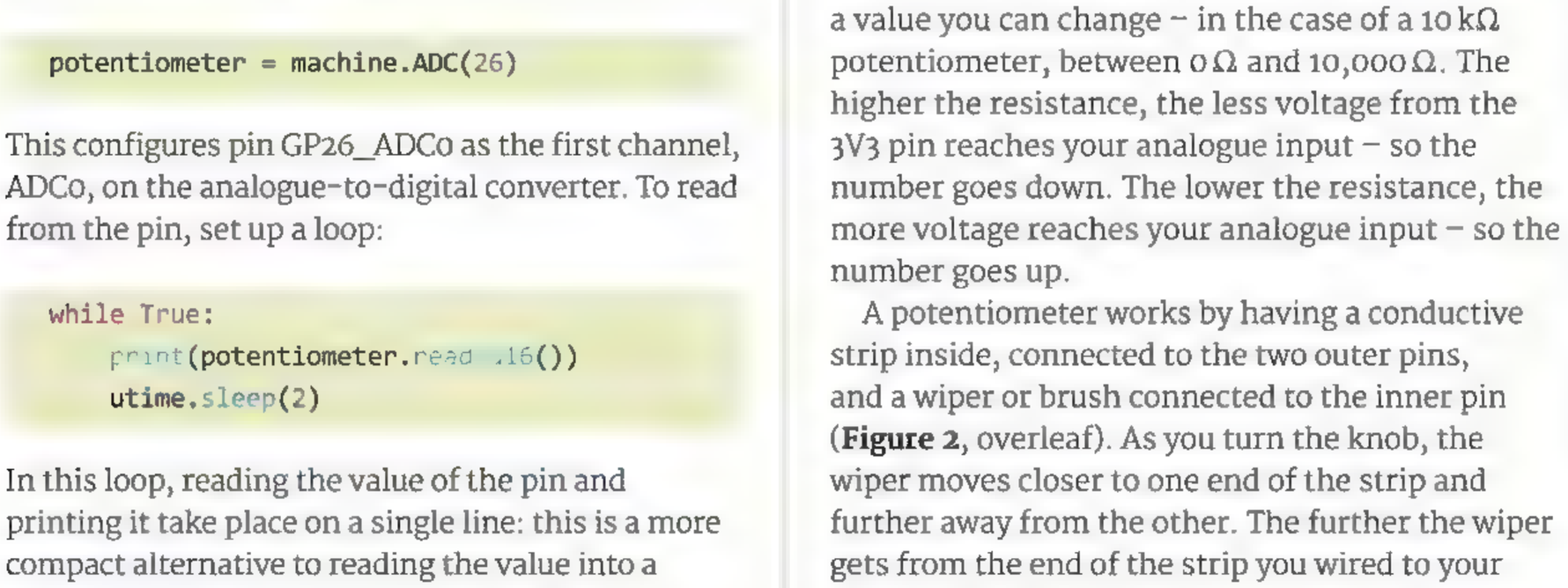


Figure 1 A potentiometer wired with two pins connected

Varistors are extremely useful components, but there's a drawback: you'll notice no matter how far you turn the knob in either direction, you can never get a value of 0 – or anywhere close to it. That's because a 10 kΩ resistor isn't strong enough to drop the 3V3 pin's output to 0 V. You could look for a bigger potentiometer with a higher maximum resistance, or you could simply wire your existing potentiometer up as a voltage divider.

A potentiometer as a voltage divider

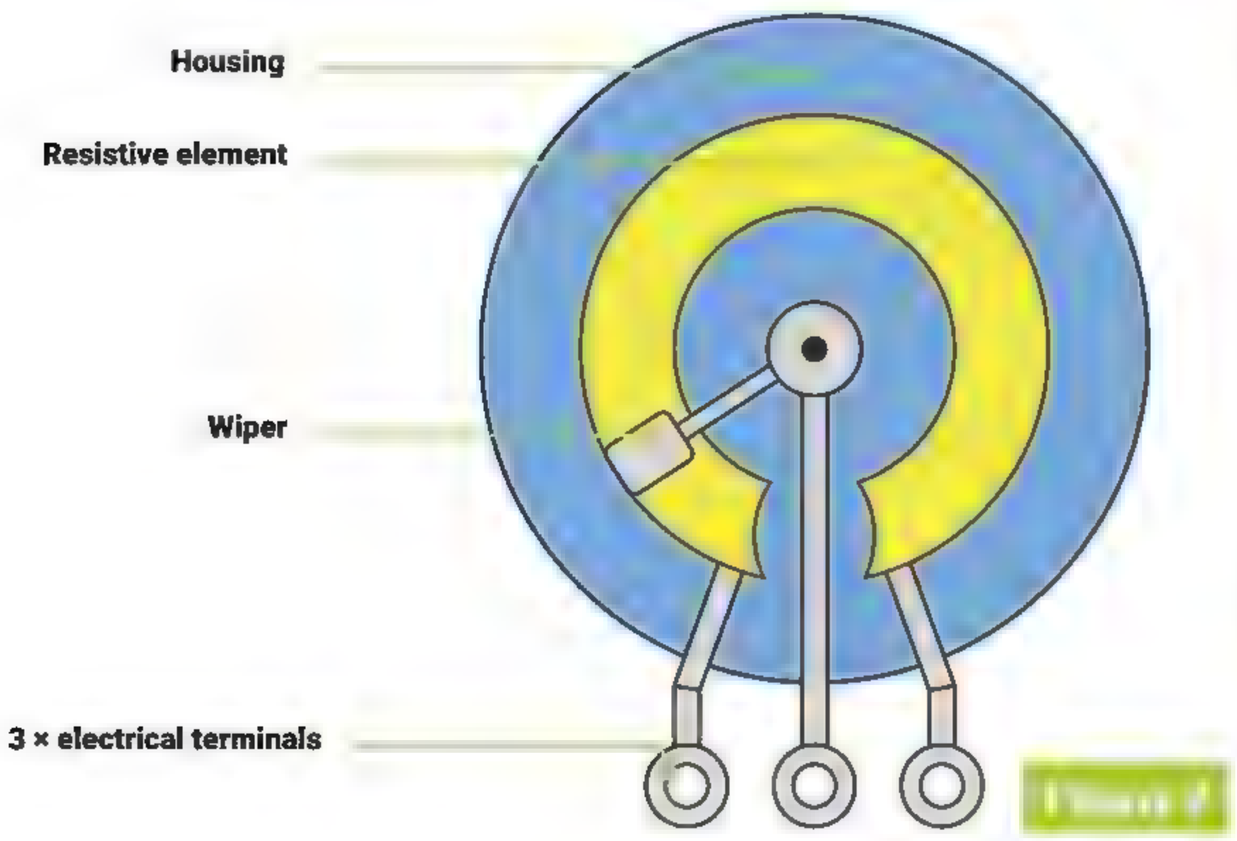
The unused pin on your potentiometer isn't there for show: adding a connection to that pin to your circuit completely changes how the potentiometer works. Click the Stop icon to stop your program, and grab two male-to-male (M2M) jumper wires. Use one to connect the unused pin of your potentiometer to your breadboard's ground rail, as shown in **Figure 3**. Take the other and connect the ground rail to a GND pin on your Pico.

Click the Run icon to restart your program. Turn the potentiometer knob again, all the way one direction then all the way the other. Watch the values that are printed to the Shell area: unlike before, they're now going from near-zero to nearly a full 65,535 – but why?

Adding the ground connection to the other end of the potentiometer's conductive strip has created a voltage divider: whereas before the potentiometer was simply acting as a resistor between the 3V3 pin and the analogue input pin, it's now dividing the voltage between the 3.3 V output by the 3V3 pin and the 0 V of the GND pin. Turn the knob fully one direction, you'll get 100 percent of the 3.3V; turn it fully the other way, 0 percent.

> You could wire your existing potentiometer up as a voltage divider

Figure 2 How a potentiometer works



The number you see printed to the Shell is a decimal representation of the raw output of the analogue-to-digital converter – but it's not the friendliest way to see it, especially if you forget that 65,535 means 'full voltage'.

There's an easy way to fix that, though: a simple mathematical equation. Go back to your program, and add the following above your loop:

```
conversion_factor = 3.3 / (65535)
```

This sets up a mathematical way to convert the number that the analogue-to-digital converter gives you into a fair approximation of the actual voltage it represents. The first number is the maximum possible voltage that the pin can expect: 3.3V, from your Pico's 3V3 pin; the second number is the maximum value the analogue input reading can be, 65,535.

Taken all together, the conversion factor is a number created by '3.3 divided by 65,535' – the maximum possible voltage divided by the range of values the analogue-to-digital converter reports, which is in turn a feature of its resolution in bits. With your conversion factor set up, you simply need to use it in your program. Go back to your loop, and edit it to read:

```
while True:
    voltage = potentiometer.read_u16() *
conversion_factor
    print(voltage)
    utime.sleep(2)
```

The first line inside the loop takes a reading from the potentiometer via the analogue input pin, and multiplies it – the * symbol – by the conversion factor you set up earlier in the program, storing the result as the variable voltage. That variable is then printed to the Shell, in place of the raw reading you used earlier.

Your finished program will look like this:

```
import machine
import utime

potentiometer = machine.ADC(26)
```

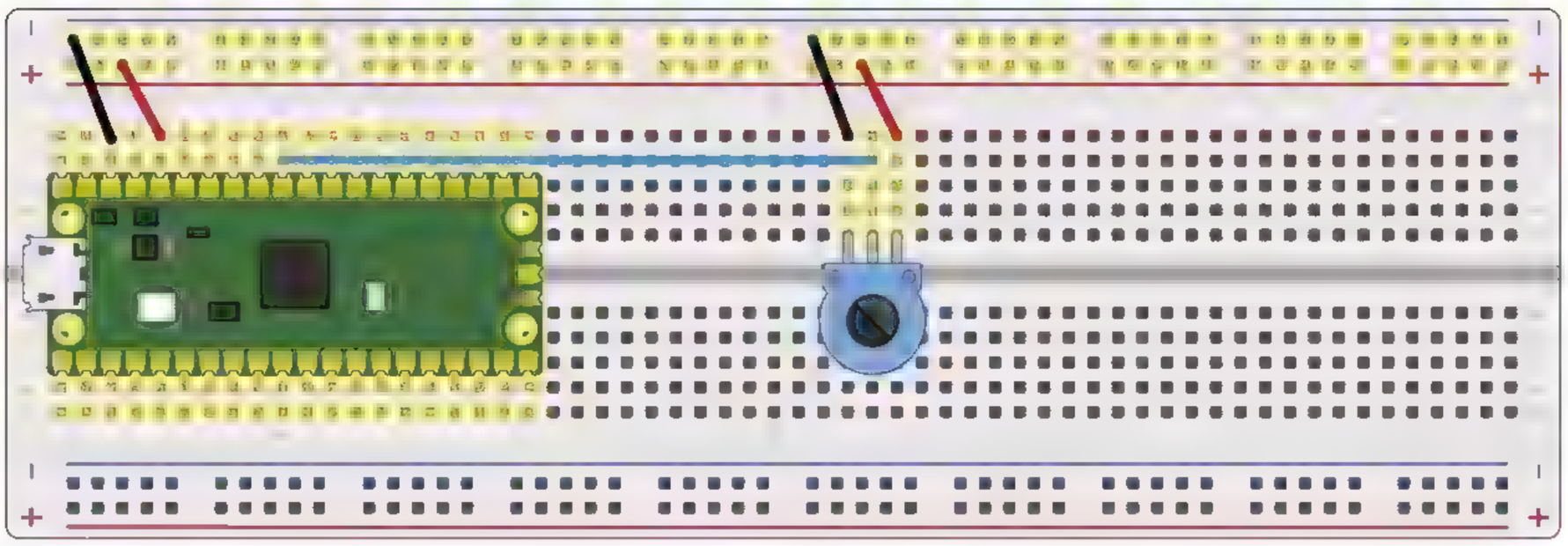


Figure 3 Wiring the potentiometer as a voltage divider

```
conversion_factor = 3.3 / (65535)

while True:
    voltage = potentiometer.read_u16() *
conversion_factor
    print(voltage)
    utime.sleep(2)
```

Click the Run icon. Turn the potentiometer all the way in one direction, then the other. Watch the numbers being printed to the Shell area: you'll see that when the potentiometer is all the way one way, the numbers get very close to zero; when it's all the way the other way, they get very close to 3.3. These numbers represent the actual voltage being read by the pin – and as you turn the knob of the potentiometer, you're dividing the voltage smoothly between minimum and maximum, 0V to 3.3V.

Congratulations: you now know how to wire a potentiometer as both a varistor and a voltage divider, and how to read analogue inputs as both a raw value and a voltage!

Measuring temperatures

Your Raspberry Pi Pico's RP2040 microcontroller has an internal temperature sensor, which is read on the fourth analogue-to-digital converter channel. Like the potentiometer, the output of the sensor is a variable voltage: as the temperature changes, so does the voltage.

Start a new program, and import the machine and utime libraries:

```
import machine
import utime
```

Set up the analogue-to-digital converter again, but this time rather than using the number of a pin, use the channel number connected to the temperature sensor:

```
sensor_temp = machine.ADC(4)
```

You'll need your conversion factor again, to change the raw reading from the sensor into a voltage value, so add that:

```
conversion_factor = 3.3 / (65535)
```

Then set up a loop to take readings from the analogue input, apply the conversion factor, and store them in a variable:

```
while True:
    reading = sensor_temp.read_u16() *
conversion_factor
```

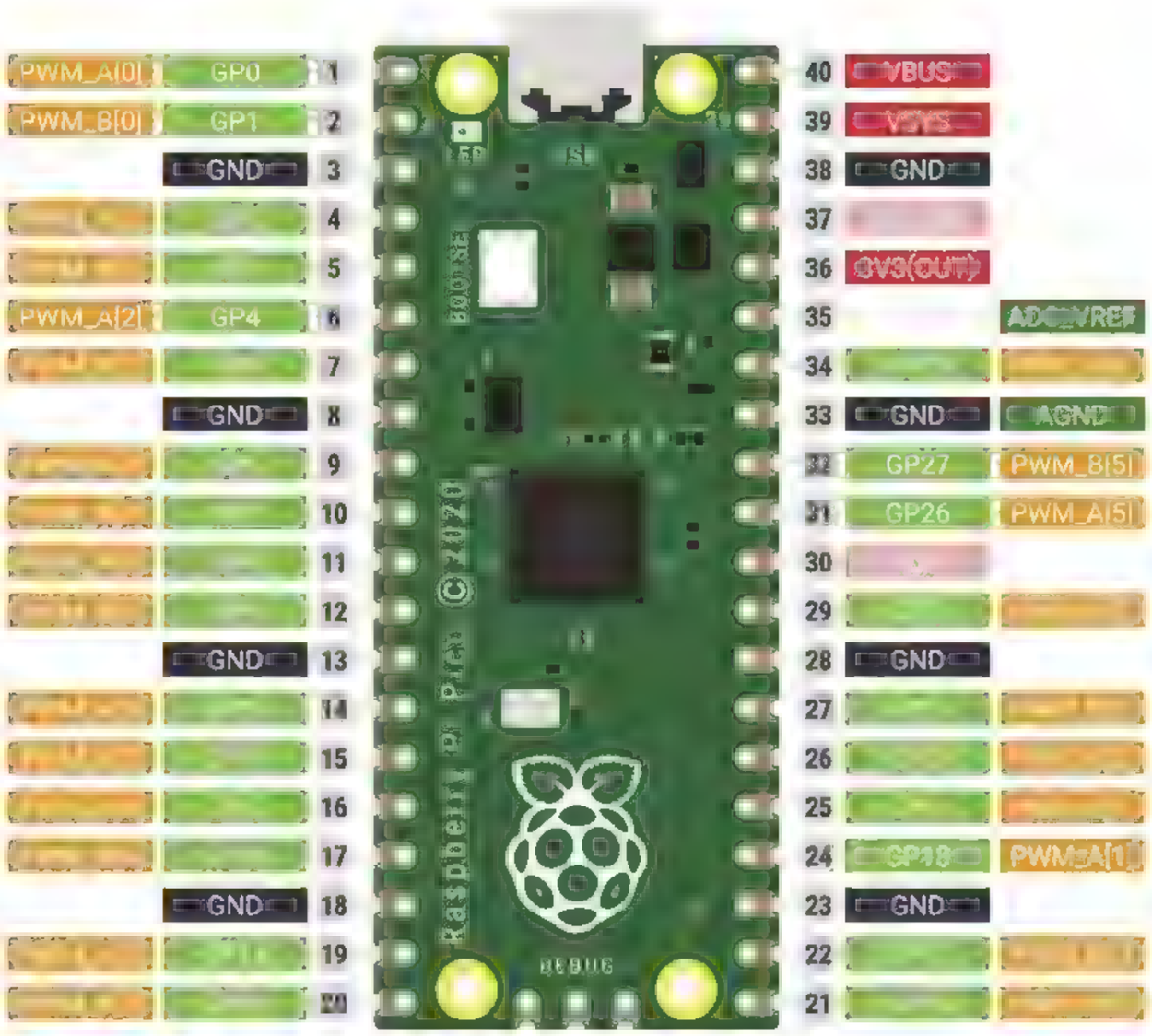
Rather than print the reading directly, though, you need to do a second conversion – to take the voltage reported by the analogue-to-digital converter and convert it into degrees Celsius:

```
temperature = 27 - (reading -
0.706)/0.001721
```

This is another mathematical equation, and one which is specific to the temperature sensor in RP2040 . The values are taken from a technical document called a data sheet or data book: all electronic components have a data sheet, which is normally available on request from the manufacturer. You can view RP2040's data sheet in the Pico documentation at **rptl.io/rp2040-get-started** – it's packed full of information on how the microcontroller works, though it's aimed at engineers so is deeply technical.

Finally, finish your loop:

```
print(temperature)
utime.sleep(2)
```



▲ **Figure 4** The pulse-width modulation pins

converts it to a digital signal the microcontroller can understand. If you want to go the other way, and have your digital microcontroller create an analogue output, you'd normally need a digital-to-analogue converter (DAC) – but there's a way to 'fake' an analogue signal, using something called pulse-width modulation or PWM.

A microcontroller's digital output can only ever be on or off, 0 or 1. Turning a digital output on and off is known as a pulse and by altering how quickly the pin turns on and off you can change, or modulate, the width of these pulses – hence 'pulse-width modulation'.

Every GPIO pin on your Pico is capable of pulse-width modulation, but the microcontroller's pulse-width modulation block is made up of eight slices, each with two outputs. Look at **Figure 4**: you'll see that each pin has a letter and a number in the orange labels. The number represents the PWM slice connected to that pin; the letter represents which output of the slice is used.

Your program will now look like this:

```
import machine
import utime

sensor_temp = machine.ADC(4)

conversion_factor = 3.3 / (65535)

while True:
    reading = sensor_temp.read_u16() *
conversion_factorwhile True:
    temperature = 27 - (reading -
0.706)/0.001721
    print(temperature)
    utime.sleep(2)
```

Click the Run icon and save your program as **Temperature.py**. Watch the Shell area: you'll see numbers being printed which represent the temperature reported by the sensor in degrees Celsius.

Try gently pressing the tip of your finger to RP2040, the largest black chip in the middle of your Pico, and holding it there: the warmth of your finger should make the chip warmer, and the temperature will rise. Remove your finger from the chip, and the temperature will fall again.

Congratulations – you've turned your Pico into a thermometer!

Fading an LED with PWM

The analogue-to-digital converter in your Pico only works one way: it takes an analogue signal and

> There's a way to 'fake' an analogue signal, using pulse-width modulation

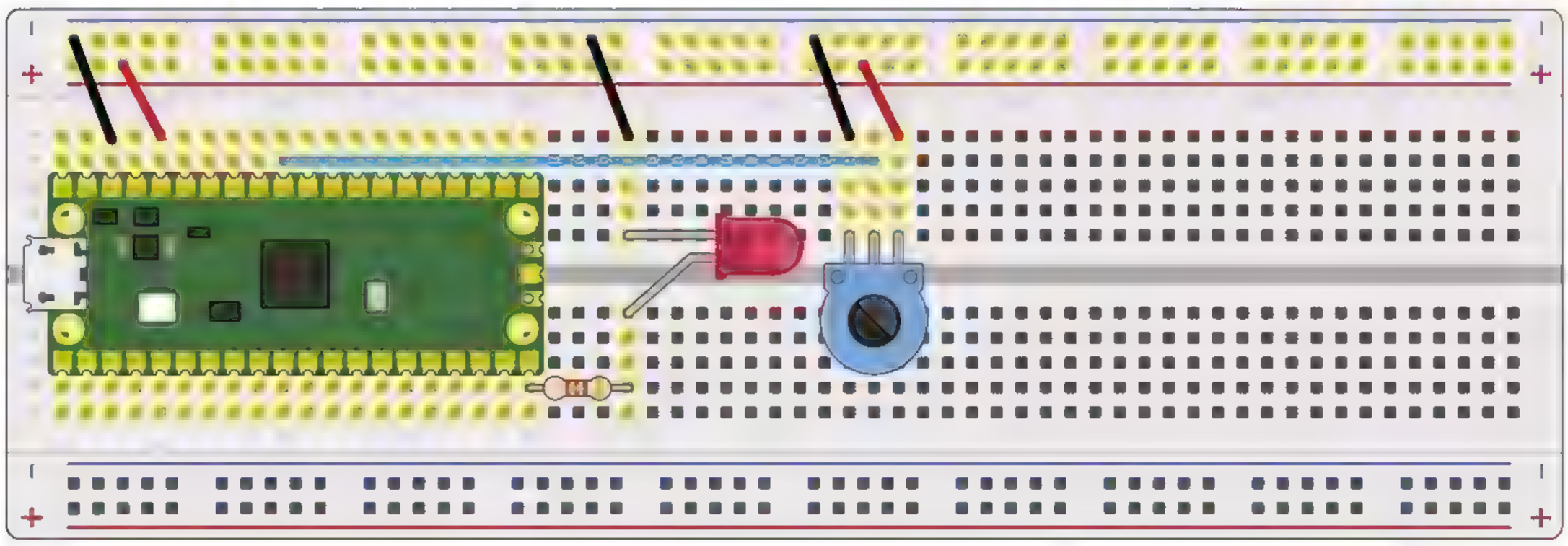
If that sounds confusing, don't worry: all it means is that you need to make sure you keep track of the PWM slices and outputs you're using, making sure to only connect to pins with a letter and number combination you haven't already used. If you're using PWM_A[0] on pin GP0 and PWM_B[0] on pin GP1, things will work fine, and will continue to work if you add PWM_A[1] on pin GP2; if you try to use the PWM channel on pin GP0 and pin GP16, though, you'd run into problems as they're both connected to PWM_A[0].

Take an LED and a 330Ω current-limiting resistor, and put them in the breadboard as shown in **Figure 5**. Wire the longer leg of the LED, the anode, to pin GP15 via the 330Ω resistor, and wire the shorter leg to the ground pin of your Pico.

Go back to your first program by clicking on its tab just under Thonny's toolbar; if you'd already closed it, click the Open icon and load **Potentiometer.py** from your Pico. Just under where you set up the potentiometer as an analogue-to-digital input, type:

```
led = machine.PWM(machine.Pin(15))
```

DOWNLOAD THE FULL CODE:
magpi.cc/github



▲ **Figure 5** Adding an LED

This creates an LED object on pin GP15, but with a difference: it activates the pulse-width modulation output on the pin, channel B[7] – the second output of the eighth slice, counting from zero.

You'll also need to set the frequency, one of the two values you can change to control, or modulate, the pulse width. Add another line immediately below reading:

```
led.freq(1000)
```

This sets a frequency of 1000 hertz – one thousand cycles per second. Next, go to the bottom of your program and delete the `print(voltage)` and `utime.sleep(2)` lines before adding the following, remembering to keep it indented by four spaces so it forms part of the nested code within the loop:

```
    led.duty_u16(potentiometer.read_u16())
```

This line takes a raw reading from the analogue input connected to your potentiometer, then uses it as the second aspect of pulse-width modulation: the duty cycle. The duty cycle controls the pin's output: a 0 percent duty cycle leaves the pin switched off for all 1000 pulses per second, and effectively turns the pin off; a 100 percent duty cycle leaves the pin switched on for all 1000 pulses per second, and is functionally equivalent to just turning the pin on as a fixed digital output; a 50 percent duty cycle has the pin on for half the pulses and off for half the pulses.

To make it so you can properly control the LED's brightness, you need to map the value from the analogue input to a range the PWM slice can understand. The best way to do this is to tell MicroPython that you're passing the duty cycle value as an unsigned 16-bit integer, the same number format as you receive from your Pico's analogue input pin. This is achieved with the use of `led.duty_u16`.

Your finished program will look like this:

```
import machine
import utime

potentiometer = machine.ADC(26)
led = machine.PWM(machine.Pin(15))
led.freq(1000)

while True:
    led.duty_u16(potentiometer.read_u16())
```

Click the Run icon and try turning the potentiometer all the way one way, then all the way the other. Watch the LED: this time, unless you're using a logarithmic potentiometer, you'll see the LED's brightness change smoothly from completely off at one end of the potentiometer knob's limit to fully lit at the other.

Congratulations: you've not only mastered analogue inputs, but you can now create the equivalent to an analogue output using pulse-width modulation!

BIG BUILDS: Handheld Builds

Complex projects that won't take up an entire room

Last month we wrote about some truly big builds. Legit furniture for your home powered by a Raspberry Pi. That's not the only way we define big builds here – they're builds that take time and generally have a lot of parts you're building yourself.

With that in mind, we decided to put together a part two with some smaller big builds. Satisfying, yet easy to store. Let's build.

ISSUE 55

magpi.cc/55

HANDHELD CONSOLE

URL: MAGPI.CC/PIGRRL

SKILLS: 3D PRINTING, SOLDERING/ELECTRONICS

LEVEL: INTERMEDIATE

Gaming on the go with this small big build

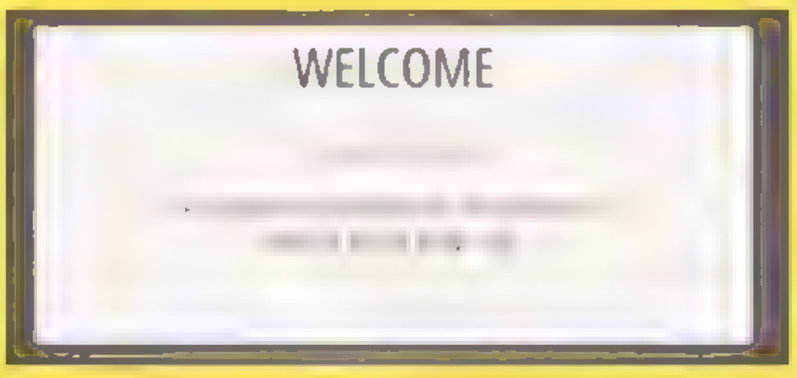
Arcade machines are great, but they don't really fit in your pocket – we've tried. While it may not seem quite as complex, making yourself a custom handheld still requires a bit of work. Luckily, we have just the build for you

01 Get your parts

There are some handy kits you can get to help build your handheld console (like from here: **magpi.cc/pigrrl**), but these only work with this specific 3D-printed case. If you're on Thingiverse looking at different cases, make sure to get the right parts.

02 Prepare a Raspberry Pi

Similarly to a full arcade machine, RetroPie really is the best choice for a handheld gaming system. You may need to install extra software for buttons attached over GPIO, and there are details on how to do this on the full guides.

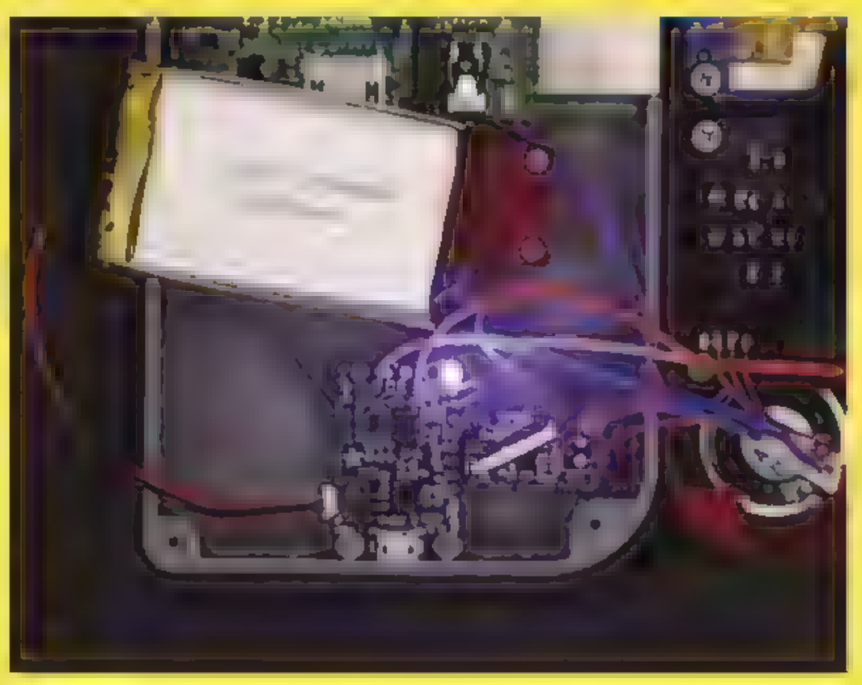


MATERIALS

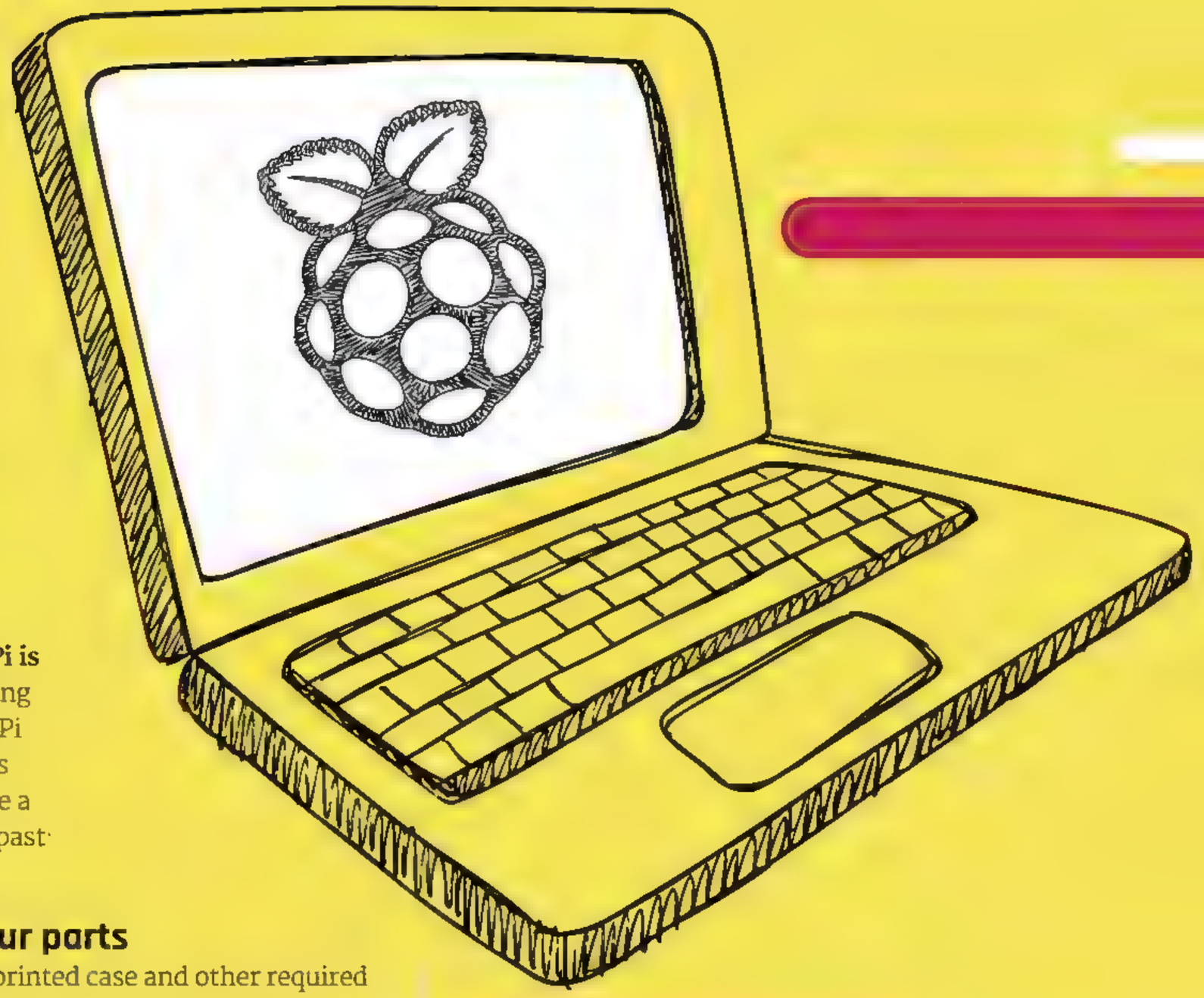
- 3D-printed case
- 2.8" PiTFT Plus
- PiGRRL Gamepad PCB
- PowerBoost 1000c
- 2500mAh battery
- PAM8302 2.5W audio amp
- Mini Metal Speaker
- 40-pin GPIO ribbon cable
- Slide switch
- tactile buttons
- 2 by 20-pin IDC box header

03 Put it together

Once you have all your parts and Raspberry Pi set up, you can put them all together. Be careful when placing wires so as not to block anything, and be gentle with any screws as some 3D-printed plastics may damage if you're a bit rough with a screwdriver.



LAPTOP

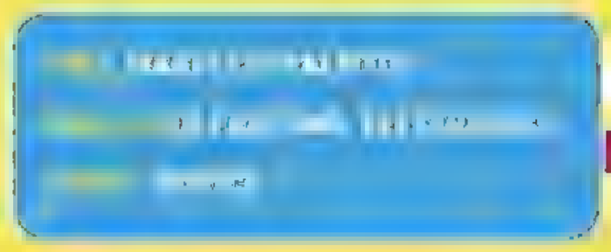


Portable computing made easy with a tiny computer

Making a laptop powered by Raspberry Pi is remarkable easy. Whether you're making it as small as possible with a Raspberry Pi Zero, or taking advantage of a Raspberry Pi 400's full set of features, there are many ways to create a Raspberry Pi laptop. Here's one we made in the past.

01 Gather your parts

Get your 3D-printed case and other required parts ready. If you're going for an alternate 3D-printed case, make sure to get the right parts for that specific build. There's very little wiring required here as the keyboard/mouse is wireless, and the screen sits on top of the GPIO pins on Raspberry Pi.



02 Set up Raspberry Pi

For easier setup, installing Raspberry Pi OS and connecting all the Bluetooth/wireless devices and networks before assembling means you can fix any issues before any screws are tightened. It can also make SD card access a lot easier. You'll need to add the software to run the mini display here as well.

03 Construct the laptop

Once your Raspberry Pi is ready, you can start assembling everything. Even though this is an easier build, make sure to be careful with wire placement and screws. Give it a test with everything in place to make sure you're happy, and then marvel at your tiny build.

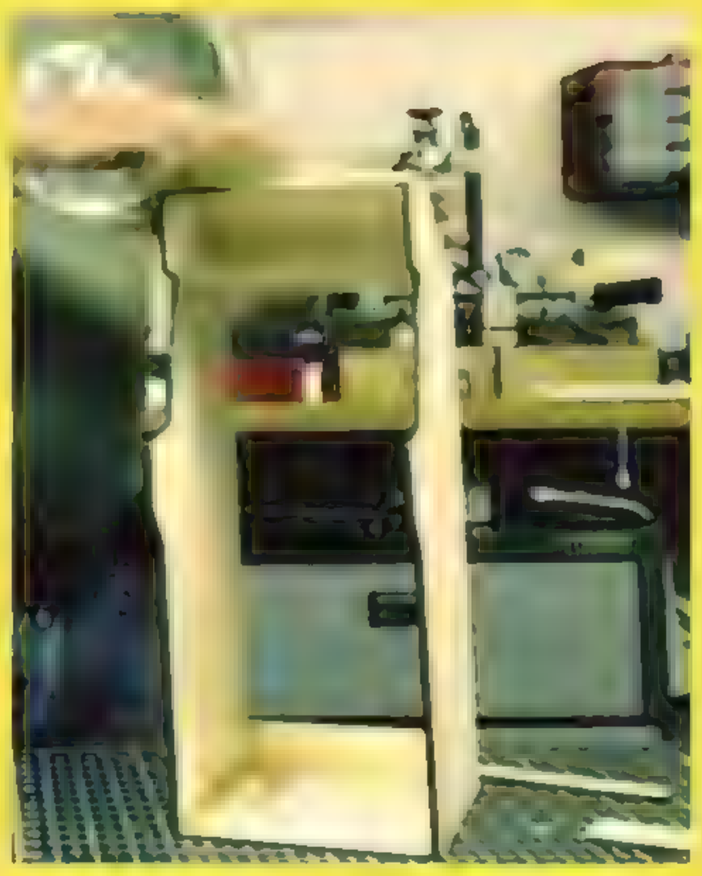
GALAGA PI

Get the arcade experience without losing space

We recently wrote about a ¾-sized replica of a Star Wars arcade cabinet, which was still pretty big. Galaga Pi, on the other hand, will sit nicely on a shelf Maker Tiburcio de la Carcova has made many small, yet working, replicas. Here's how to do it:

01 Cut and print

Parts modelled in Sketchup (**sketchup.com**) are laser-cut on 3 mm and 6 mm plywood for the main frame. Other parts are 3D-printed, such as the coin slot, and there's acrylic and aluminium to complete Tiburcio's requirement.

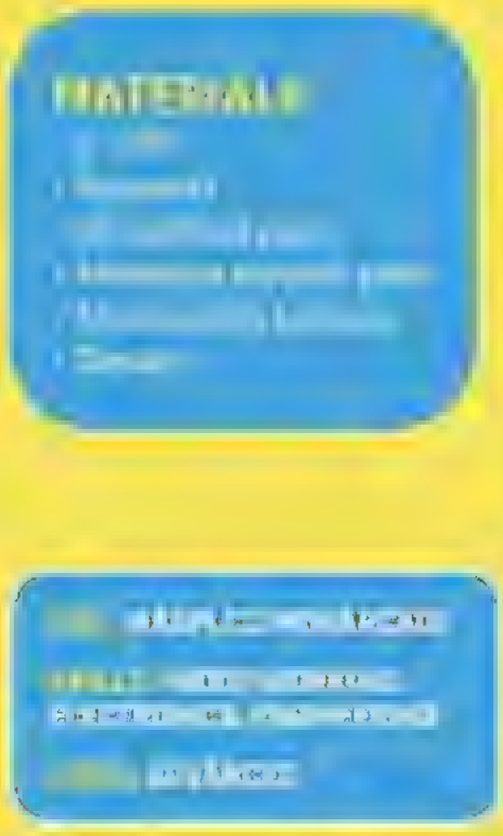


02 Improvise

"The joystick was a challenge, since there's no joystick of that scale that looked accurate," Tiburcio says. "I started with a model I found at Thingiverse (**thingiverse.com**) and improved [it] with a metal stick and a spring system "

03 Construct

Carefully put all the parts into the case, apply the decals, and top with a clear coat for a painted look Close up the case and you have yourselves a mini arcade cabinet ready for Galaga, Space Invaders, and many other classics.



Warning!
CRT monitor

Be careful working with CRT monitors as they can hold high levels of electrical charge

magpi.cc/crt

PIP-BOY

A recycling project that will make you the pride of the wasteland

Raspberry Pi Pip-Boy builds are not at all uncommon – something about a cross-section of nerds and the fact it has 'Pi' in its name probably helps. We particularly like this version that uses scrap to make one from scratch.

01 Prototype

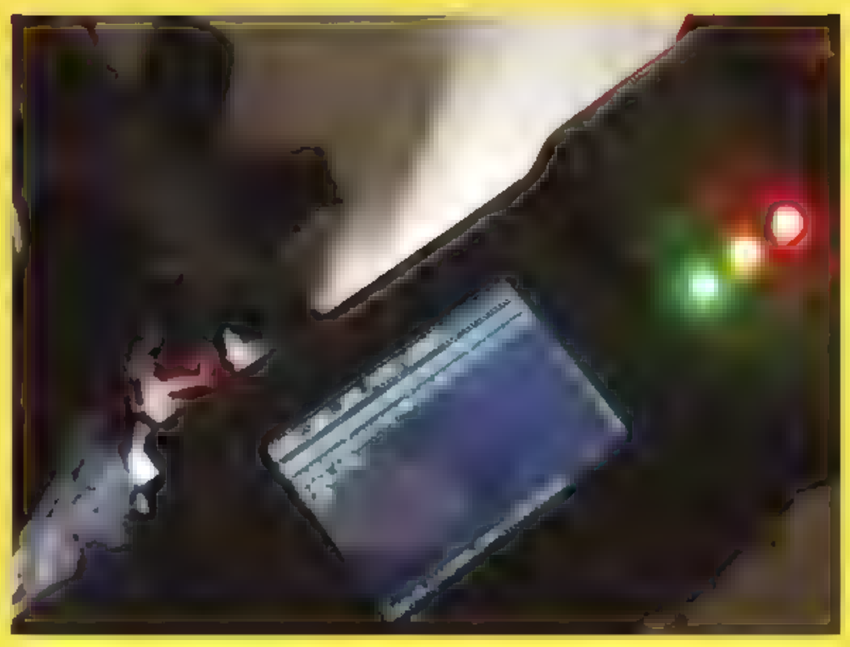
As you're making a lot of parts from scratch, prototyping with cardboard or paper on each step is a great way to make sure your patterns and cuts and modifications will work in real life Remember: gluing stuff back together ranges from 'a big pain' to 'impossible'

02 Test

As with most projects, you should probably get your Raspberry Pi and custom software up and running before installing it Testing as you go is always a great way to fix mistakes before you seal something into a case

03 Wear

If you plan to wear it for a costume or at an event, do a trial run around your home during a normal day. You'll quickly learn the best way to wear it without either damaging it or other things around your home, and find out how dexterous you still are.

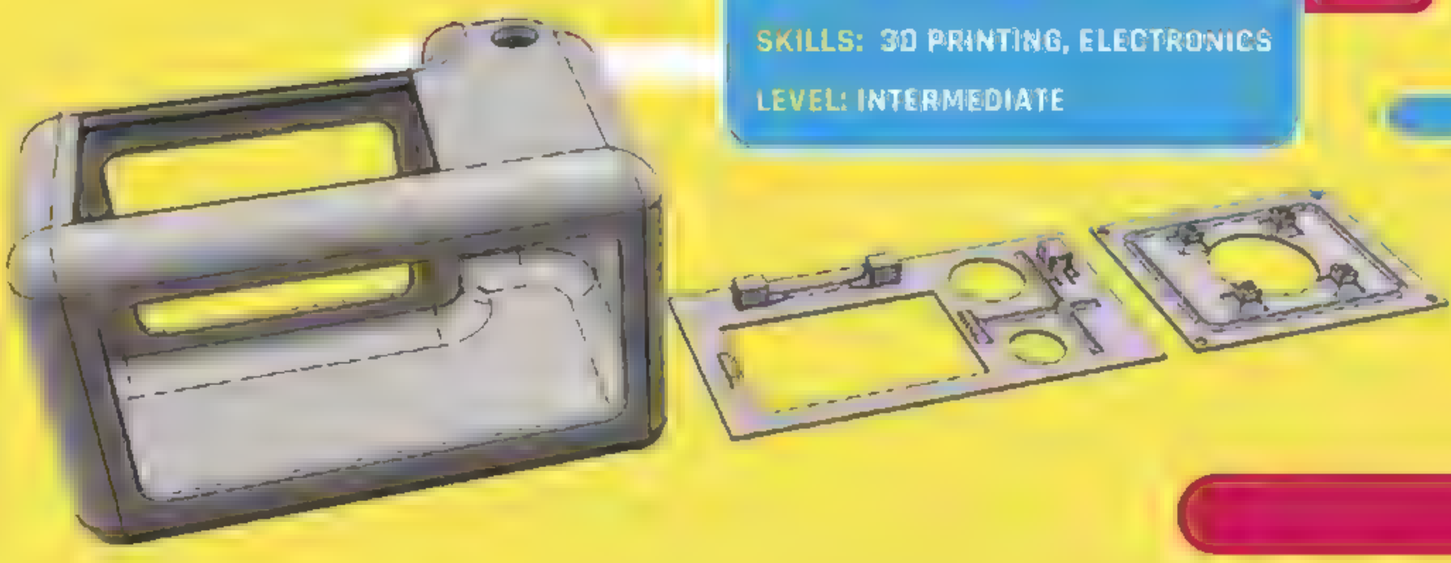


POLAPI-ZERO

Bring back instant cameras with this tiny yet mighty build

URL: MAGPI.CC/POLAPI
SKILLS: 3D PRINTING, ELECTRONICS
LEVEL: INTERMEDIATE

While this a little different from your classic instant camera, it remains functionally the same and it's honestly a lot cooler as well. A thermal printer – the kind you'd find for printing receipts – prints photos captured by a Raspberry Pi camera. It even has a live view to line up shots



01 Print

This project uses a custom 3D-printed camera chassis that works using the exact parts listed. It will fit the classic Raspberry Pi Camera Module, the buttons, and the screen, as well as a Raspberry Pi Zero to power it all.

02 Download and install

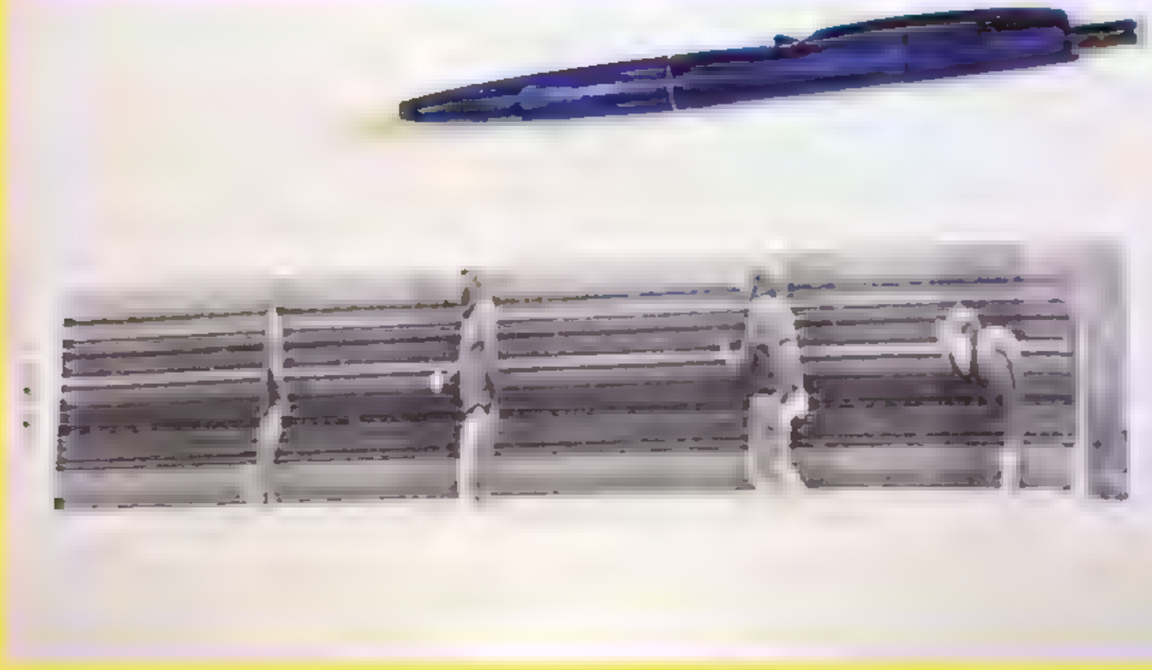
The creator, Pierre Muth, has done some great programming and image processing for this build so that the dithering of the black and white pixels looks more coherent on the final printed photo. You'll need to download the code to make it work this well

03 Assemble

Once you're done testing, carefully squeeze everything into the chassis and give it a charge. You're now ready to go out and take some weird and wonderful photos.

MATERIALS

- Nano Thermal Receipt Printer – TTL Serial
- Sharp Memory LCD LS027B7DH01A
- Raspberry Pi Camera Module
- 6 × buttons
- 3D-printed case
- 5A 3A voltage regulator
- 2S 7.2V battery



Marty the Robot V2

▶ Robotical ▶ **robotical.io** ▶ From £377 / $420

We take this two-legged robot for a walk and wave. By **Lucy Hattersley**

SPECS

PROGRAMMING LANGUAGES: Scratch, Python, JavaScript, ROS (Robotic Operating System), OpenAPI

CONNECTIVITY: I2C, Bluetooth, USB, serial, WiFi

BATTERY: 2600 mAh Li-ion rechargeable battery, 2–3 hours of use on full charge

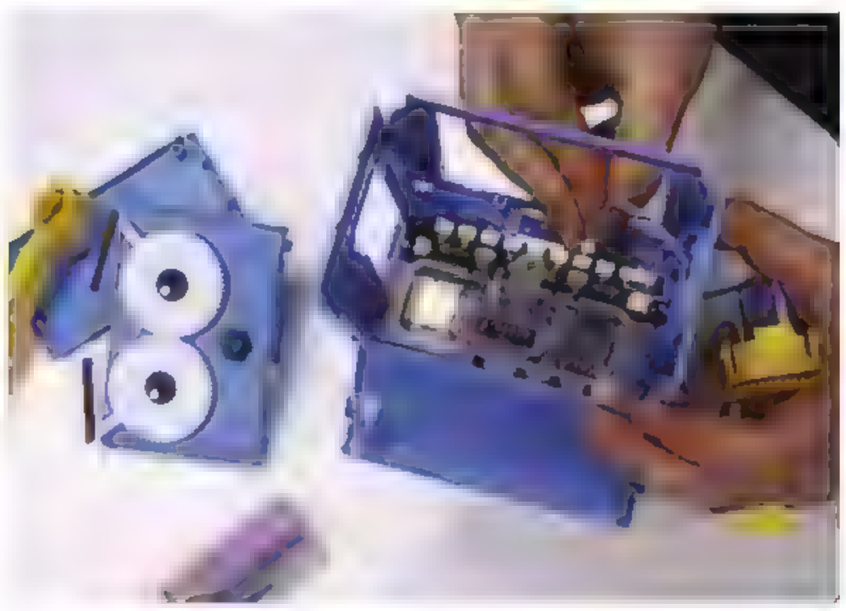
SENSORS: Accelerometer, tilt sensor, motor current sensors, motion position sensing, fall detection, IR proximity, colour & line sensor. Optional extras: distance sensor, noise sensor, compass, temperature sensor

Marty the Robot V2 is a small, personable two-legged robot that uses nine different servo motors to move around. It's capable of a wide range of small movements, including walking in both directions, rotating, side-stepping, kicking, waving, and waggling its eyes.

All of this movement is controlled by a custom RIC (Robot Interface Controller) based upon an ESP32 microcontroller (**magpi.cc/martytechspecs**). It has built-in Bluetooth, wireless LAN, an accelerometer, and a speaker. It's powered by a rechargeable battery with a built-in USB-C charger.

With his big blue head and various accessories, Marty is a good-looking piece of kit, the sort of personable toy that children can make friends with. However, there's a lot to unpack here, surprisingly so for a device that looks very much at home on the shelf of a local toy store. There is a scalable level of complex interaction that can move right from key stage 2 (7-year-old and upwards) to the university level.

Right at the entry level, you don't need any computer at all. With an optional IR colour sensor (£19, **magpi.cc/martycolour**), Marty is controlled by placing coloured cards on the floor: green to move forward, purple to slide right, and so on.



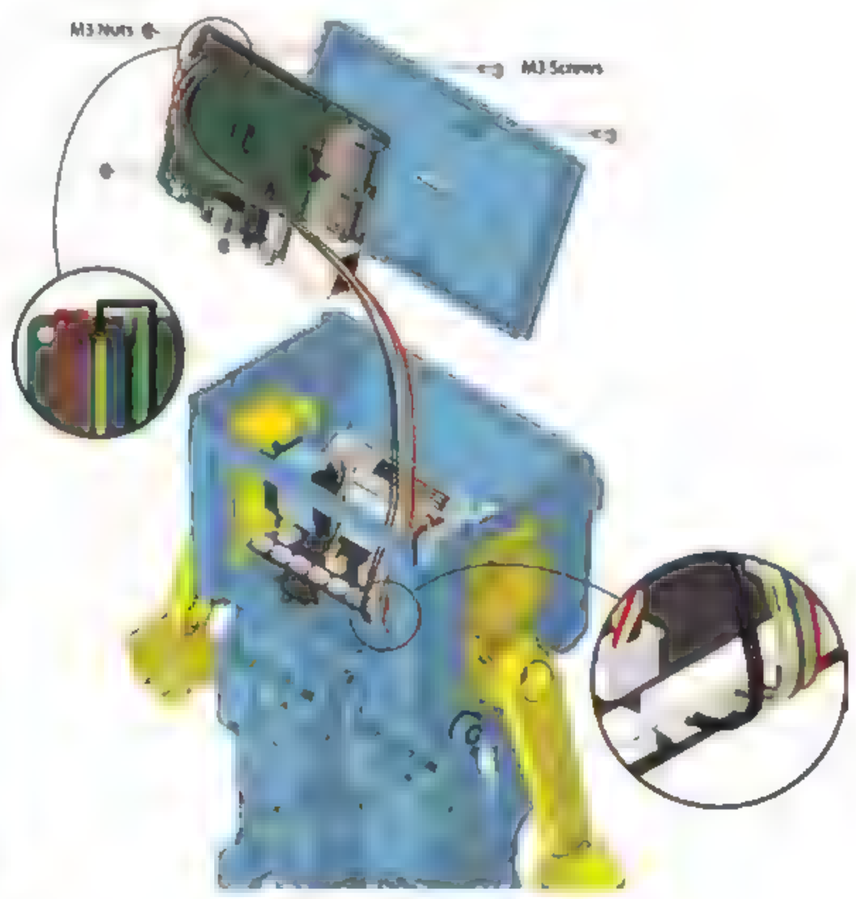
▶ Inside Marty is a RIC (Robot Interface Controller) that can connect to Raspberry Pi via a wireless network

The next stage of interaction is via an Android or iPhone app. This connects to Marty via Bluetooth and provides button controls (walk, rotate, and so forth). The app also has a built-in version of Scratch 3 that is packed with blocks to control Marty. So children can use Scratch to create programs for Marty and learn the basics of programming with a physical device (much more interactive than watching a sprite shuffle around the screen). To this end, there are a range of teacher guides and lesson plans available (**learn.martytherobot.com**).

Raspberry Pi connection

The app is also used to connect Marty to a local wireless LAN, which is where a Rtaspberry Pi can step in. Once on your network, you can move from

▼ Fitting Raspberry Pi inside Marty V2



> We found Marty personable, fun to play with, and packed with features – some of which scale surprisingly high up

Scratch to Python and connect to Marty using the MartyPy library. For example, here was our program to make Marty dance:

```
from martypy import Marty
my_marty = Marty("wifi","192.168.0.45")
my_marty.dance()
```

There's a comprehensive Knowledge Base portal on Robotical's website (**magpi.cc/roboticalknowledge**) with example code, tutorials, and a Python function reference guide.

Marty does not – by default – contain a Raspberry Pi computer. Instead, you connect to the built-in RIC (Robot Interface Controller). However, you can install a Raspberry Pi inside the head unit and connect it to the RIC (**magpi.cc/martyraspberrypi**) using the supplied cable.

Putting a Raspberry Pi inside Marty gives it independence from the network and other computers or the app. There's also a decrease in latency time, so response time is faster. There's the tantalising possibility to expand Marty's abilities with voice interaction and image processing to make the robot more like a walking smart assistant.

Even if students never make it that far, learning code by moving a physical device around, rather than shuffling sprites around on the screen, is a powerful concept. And we found Marty personable, fun to play with, and packed with features – some of which scale surprisingly high up. The detailed documentation helps get you started and gives you direction.

Verdict

A great little robot that brings Scratch and Python code to life. Don't be fooled by the cute exterior: there's a lot of potential inside this robot.

8/10

micro:Pi

▶ SB Components ▶ magpi.cc/micropi ▶ From £50 / $71

An all-in-one educational kit to teach IoT built on Raspberry Pi 3A+. By **Rob Zwetsloot**

SPECS

FEATURES:
Buzzer, I2C Grove connectors, IR sensor, 4 × NeoPixel RGB LEDs, push-buttons, OLED screen

SCREEN DETAILS:
OLED 0.96˝128×64, Colours: blue and yellow

Answering the question of 'what should I do with my Raspberry Pi' is easier than ever these days. However, sometimes there are kits you can get that already answer the question for you, much like micro:Pi. Like a pi-top or Kano product, it builds upon a Raspberry Pi to add more features right out of the box.

In this case, micro:Pi uses a Raspberry Pi 3A+ as its base, making use of the smaller footprint of the square A+ range. All the extra bits and bobs are neatly stacked on top, only increasing its vertical size in the process. It has an impressive array of components attached as well, including lights, IR sensors, buttons, audio in, and a little OLED screen to boot.

> All the extra bits and bobs are neatly stacked on top, only increasing its vertical size in the process

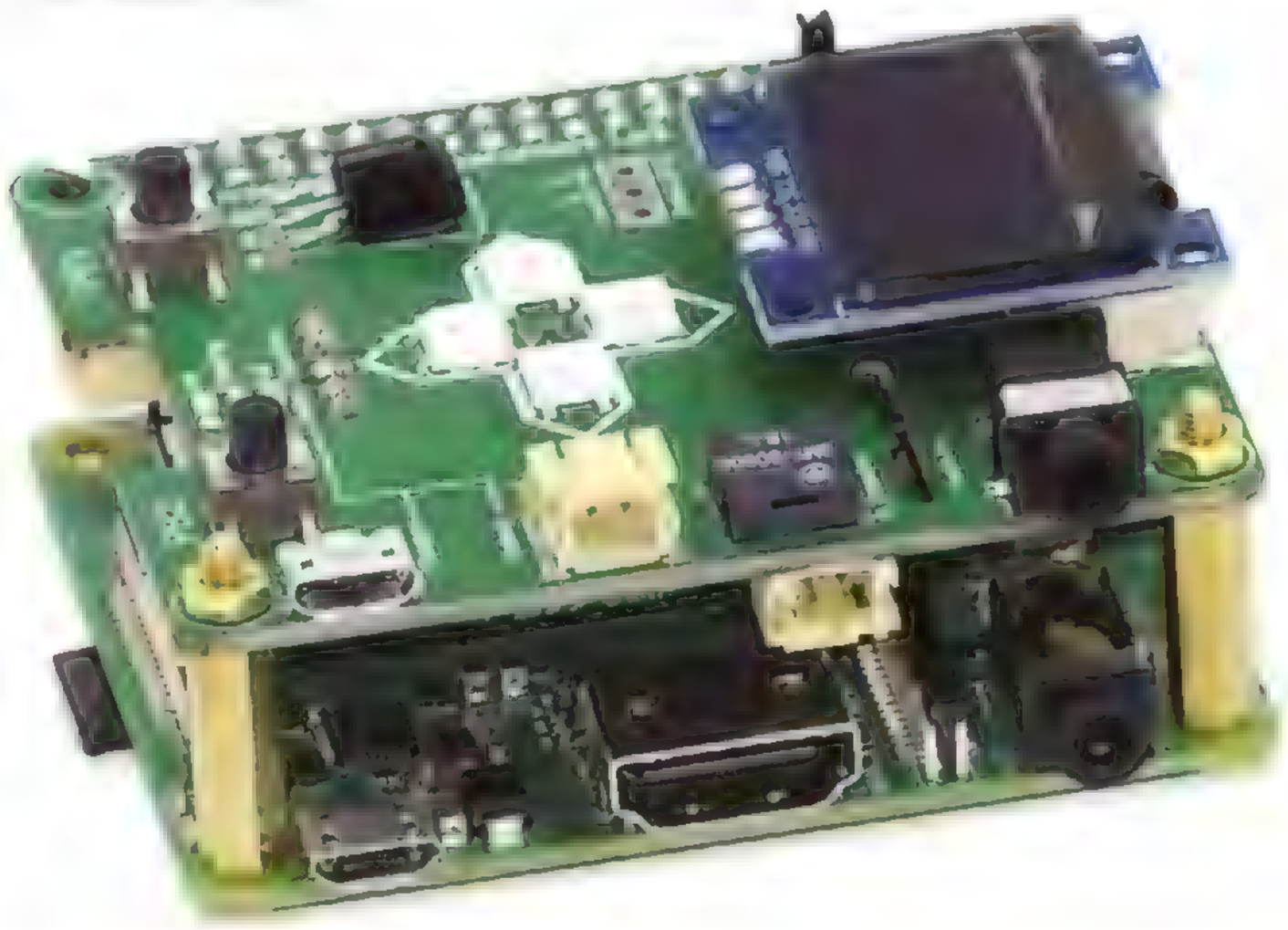
Teachable moment

Its creator, Dr Anwar Bashir, tells us that a number of functions were requested by educators – such as a self-test when an internet connection is made, and example code discreetly copied to the examples folder on each boot.

These code examples make use of the full range of functionality, with programs available to work out of the box with the buttons, LEDs, and screen, as well as functionality that can easily be added via the Grove connectors, such as motors.

The code is clearly and concisely written in Python, although it does make use of a special micropi library (similar to GPIO Zero) that will not be common elsewhere with Raspberry Pi.

Still, with all the extra functionality this adds, it's a great beginner kit that can transition to doing much more advanced stuff like robotics and home automation.



▶ The micro:Pi HAT is stacked on top of Raspberry Pi 3A+ to extend its features in a neat and tidy way

Verdict

A great starter kit with good examples that could help take new and younger coders from beginners to advanced users.

9/10

10 Amazing: Displays

See what you're doing with these varied solutions. By **Rob Zwetsloot**

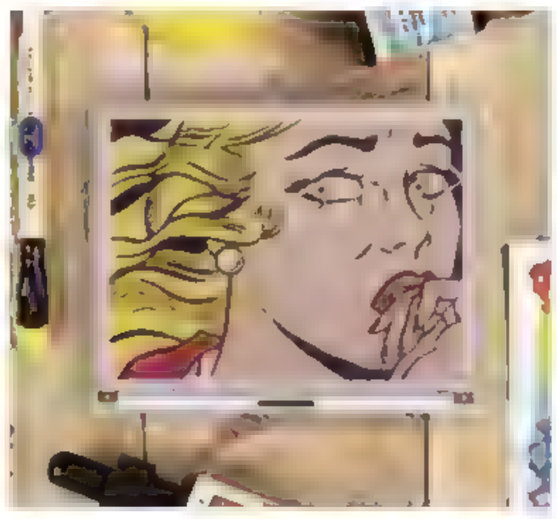
Some Raspberry Pi projects don't need a screen, and they're great. Sometimes, though, having something to see is a big part of a project. We don't just mean a standard monitor either: sometimes you need something a bit more specialist. Here are some great options to choose from.

▲ Official Raspberry Pi 7" Touchscreen Display

A great standard

The official display is a nice, fairly large screen that is used in many tablet projects, especially as it's touchscreen as well. It's 800×480 pixels and connects to the DSI Display port on Raspberry Pi.

£60 / $85 | **magpi.cc/officialdisplay**



▸ Inky Impression

Living paper

This 5.7-inch, seven-colour e-paper display is gorgeous, and perfect for projects that don't need a high refresh rate or need to look a certain way.

£66 / $94

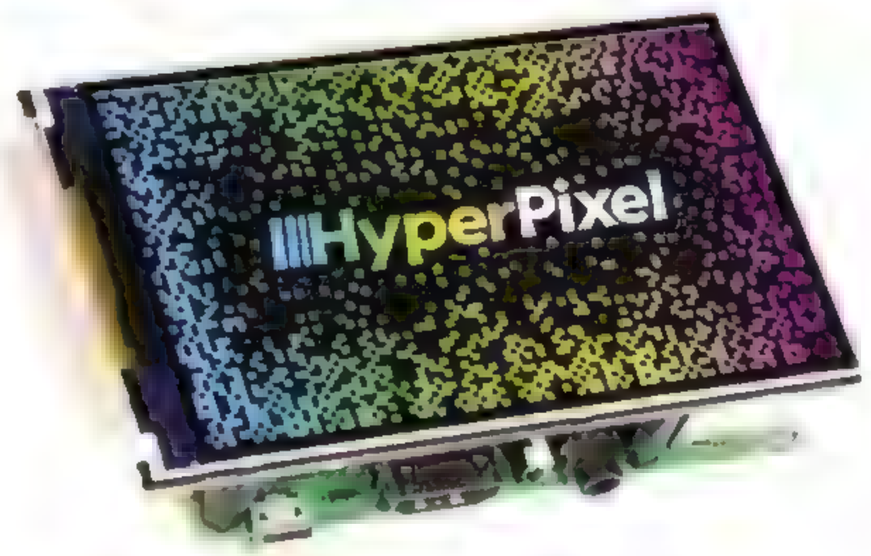
magpi.cc/inkyimpression

▲ 1.3" SPI Colour Round LCD Breakout

Your new watch

A tiny yet vibrant, circular colour display for your smallest Raspberry Pi projects, or at least the projects that need a 1.3-inch round screen.

£23 / $33 | **magpi.cc/roundlcd**



▲ HyperPixel 4.0

High-resolution display

This 800×480 display is super-reasonably priced, and also comes in a touch version. It's bright and vibrant and honestly one of the best smaller traditional screens for Raspberry Pi.

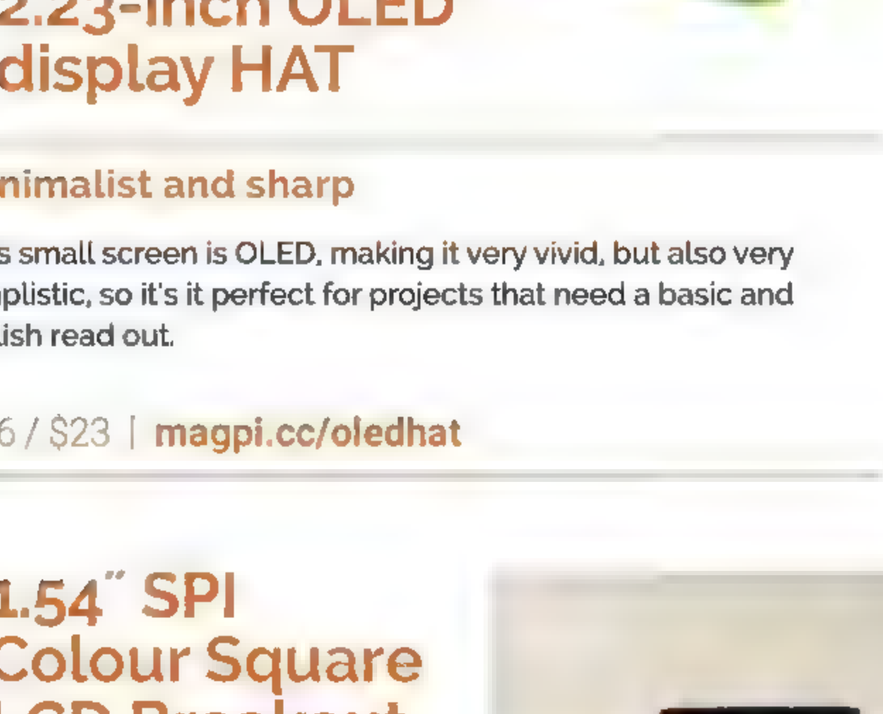
£39 / $55 | **magpi.cc/hyperpixel**

▲ PiTFT Plus

Classic display HAT

Probably the first big display add-on for Raspberry Pi, the PiTFT is still a great bit of kit. It has some physical buttons, along with touch sensitivity.

£41 / $35 | **magpi.cc/pitft**

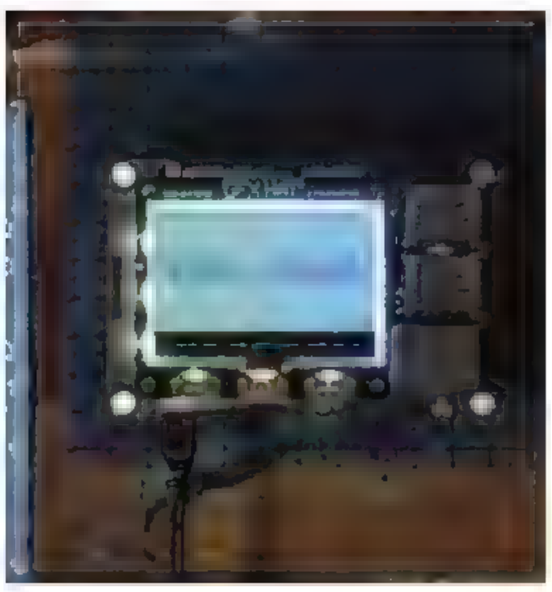


▲ 2.23-inch OLED display HAT

Minimalist and sharp

This small screen is OLED, making it very vivid, but also very simplistic, so it's it perfect for projects that need a basic and stylish read out.

£16 / $23 | **magpi.cc/oledhat**



▸ GFX HAT

Mechanical utility

A different style of display, the GFX HAT is a simpler screen that also includes nice capacitive touch buttons. Perfect for projects that need a readout and some interactivity, like a thermostat.

£23 / $32
magpi.cc/gfxhat

▲ A normal monitor

You probably have one

If you want to use Raspberry Pi as a desktop computer, there's no better way than a standard monitor. Just plug it in with an HDMI cable and off you go.

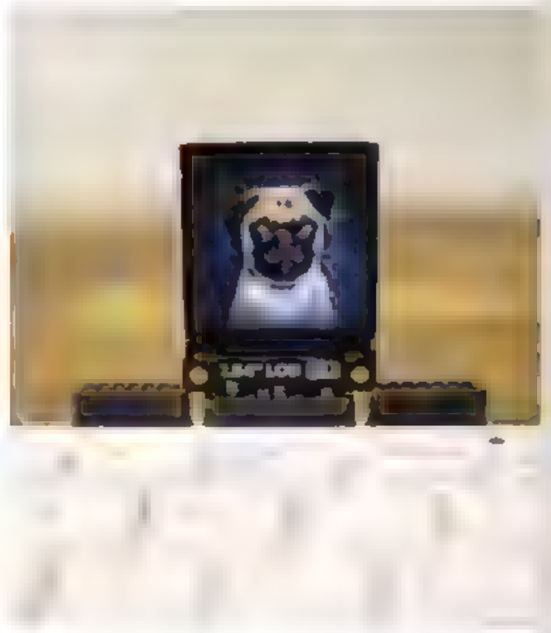
Prices vary

▸ 1.54" SPI Colour Square LCD Breakout

Four awesome sides

If a round display won't cut it, can we tempt you with its more angular sibling? It's slightly larger but still very small at 1.54 inches.

£18 / $26
magpi.cc/squarelcd



▲ 7" HDMI LCD

Tablet monitor

An alternative to the official display, with higher resolution. It connects via HDMI, so it's a bit of a more traditional monitor, although it does have its own case with a kick-stand.

£55 / $78 | **magpi.cc/hdmilcd**

Learn Linux with Raspberry Pi

Get under the hood with the world's most interesting operating system. By **Lucy Hattersley**

Work From Home

Alasdair Allan

Price
Free

magpi.cc/workfromhome

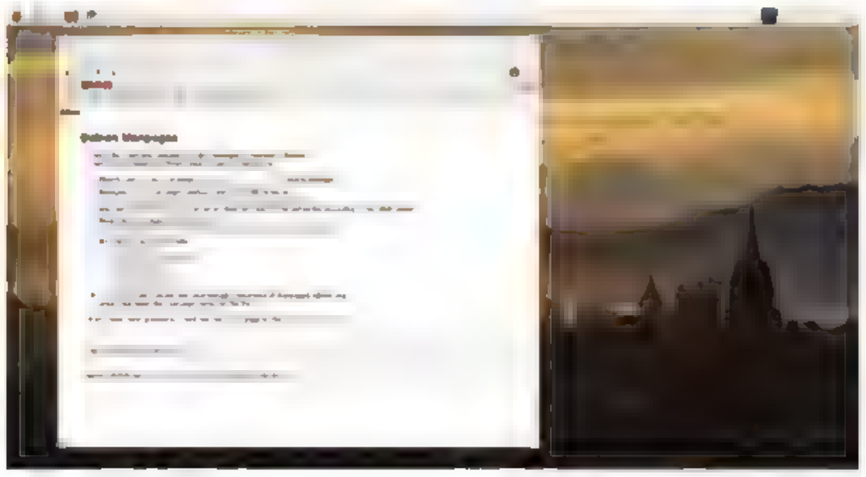
Linux is the open-source operating system that forms the basis of Raspberry Pi OS (which is based on Debian, a version of Linux). There is a baffling array of resources for learning Linux, but if you really want to get to grips with Linux and Raspberry Pi, then you have to live inside it: turn off Windows and macOS.

It's a lot like learning a second language: nothing beats the exposure of living in a place where that's the only language on offer. We've written a couple of articles about working and living with Raspberry Pi as your main computer. Take a look at *The MagPi* 93 (**magpi.cc/93**) and *The MagPi* 85 (**magpi.cc/85**), as well as this fantastic blog piece by Alasdair Allan: **magpi.cc/workfromhome**.

These articles enable you to bring Raspberry Pi up to speed as your main work computer. Spend as much time as you can in Linux – and in particular, spend as much time in the command line as possible.

Linux resources

Make sure you bookmark these pages



MAN PAGES

Man is the command to enter to learn more about Linux, and it has its own man page. Enter `man man` at the command line to read the manual.
manpages.debian.org

COMMANDLINEFU.COM

This user-voted website is a list of handy commands that you can use to master Linux. Sort by 'votes' to see some real gems.
commandlinefu.com

THE LINUX DOCUMENTATION PROJECT

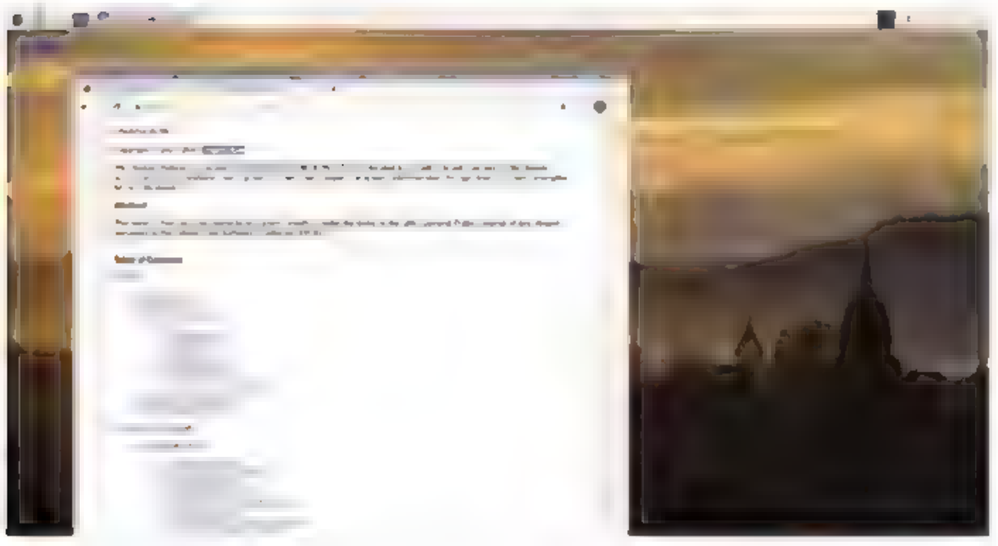
This website is a great resource for any Linux user. Packed with links to guides and how-to tutorials.
tldp.org

Debian Reference

Osamu Aoki

Price
£85 / $110
magpi.cc/debianref

Raspberry Pi OS is based on Debian (debian.org),which has a fantastic resource that's packed with information about Debian and Linux in general. You can read it online, and install it straight into Raspberry Pi OS from the command line:



```
sudo apt install debian-reference
```

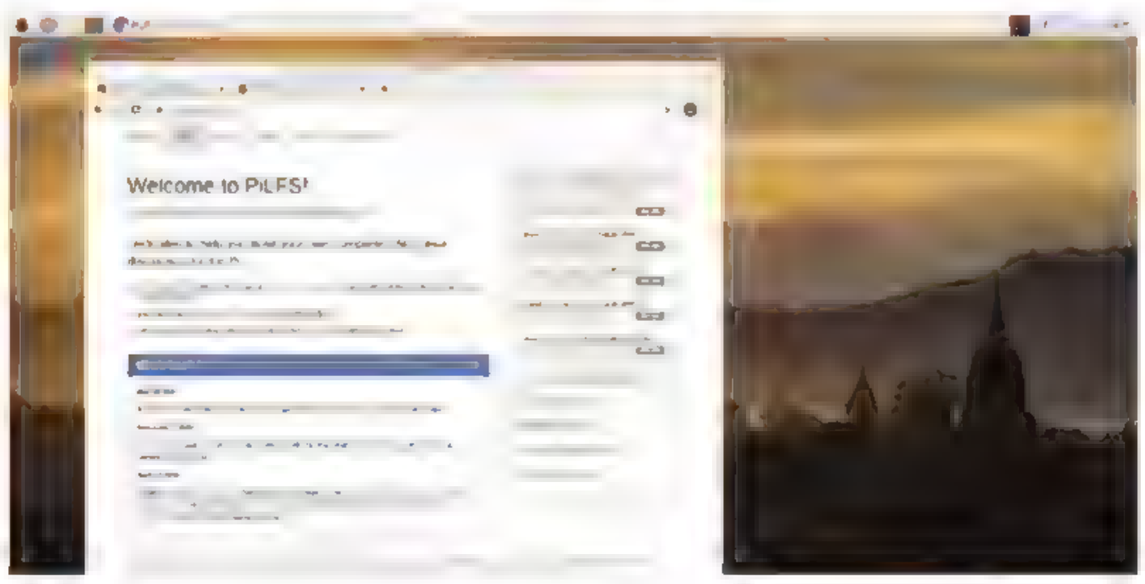
After installation, you'll find Debian Reference sitting under the Raspberry Pi Menu > Help section. It'll open in Chromium, but you can read the files offline in HTML, text, EPUB, or PDF. You can also access the files directly in **/usr/share/debian-reference**.

Debian Reference has a great selection of GNU/Linux tutorials for the beginner (covering console basics and the file system). Then it works through in-depth package management, account authentication, networking, and the X Server windowing system.

Learn Linux from Scratch

Gerard Beekmans

Price
Free
magpi.cc/lfs
magpi.cc/pilfs



Learn Linux from Scratch is a project that provides step-by-step instructions to build your own custom Linux system from, well, scratch.

The great thing about this approach is that you learn how all the different elements are used together to form an OS. You cover bringing all the main packages together, building a file system, compiling all the tools, and configuring the system to boot.

This is not a beginner's guide. It is, however, an excellent way to get a deeper understanding of Linux.

There is a 350-page book that's now on the tenth version The book is designed for AMD/Intel CPUs, but you can modify it to work with the ARM CPU, and PiLFS (**magpi.cc/pilfs**) has all the tweaks you need to build your own Linux from Scratch on Raspberry Pi.

Linux courses

Get some help with these courses

LINUX BASICS AND BASH SCRIPTING WITH RASPBERRY PI

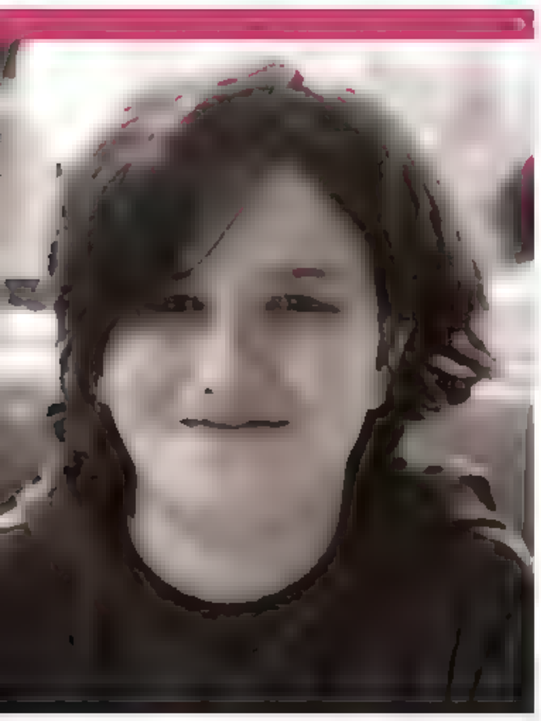
This Udemy course walks you through Linux with a Raspberry Pi and a special focus on scripting.
magpi.cc/linuxbasics

RASPBERRY PI SOC DATASHEETS

This edX course by The Linux Foundation is an online affair that you work through with other students. It's free and you can pay extra for certification.
magpi.cc/edxlinux

THE LINUX COMMANDS HANDBOOK

Learn the basics of Linux by getting to grips with the commands you'll actually use.
magpi.cc/freecodelinux



Al Pemartin

A young maker from Andorra with a broad set of skills in making and the arts. **By Rob Zwetsloot**

> Name Al Pemartin | > Occupation Student
> Community role Content creator | > URL linktr.ee/popito

On a recent Digital Making at Home stream (rpf.io/home), Al dialled in from Andorra to talk about the stuff they'd been making, from robots and video games to music and bandanas. What started with them watching their dad write code has spawned into an interest in coding and making that has resulted in this 13-year-old having a presence in several online creative communities.

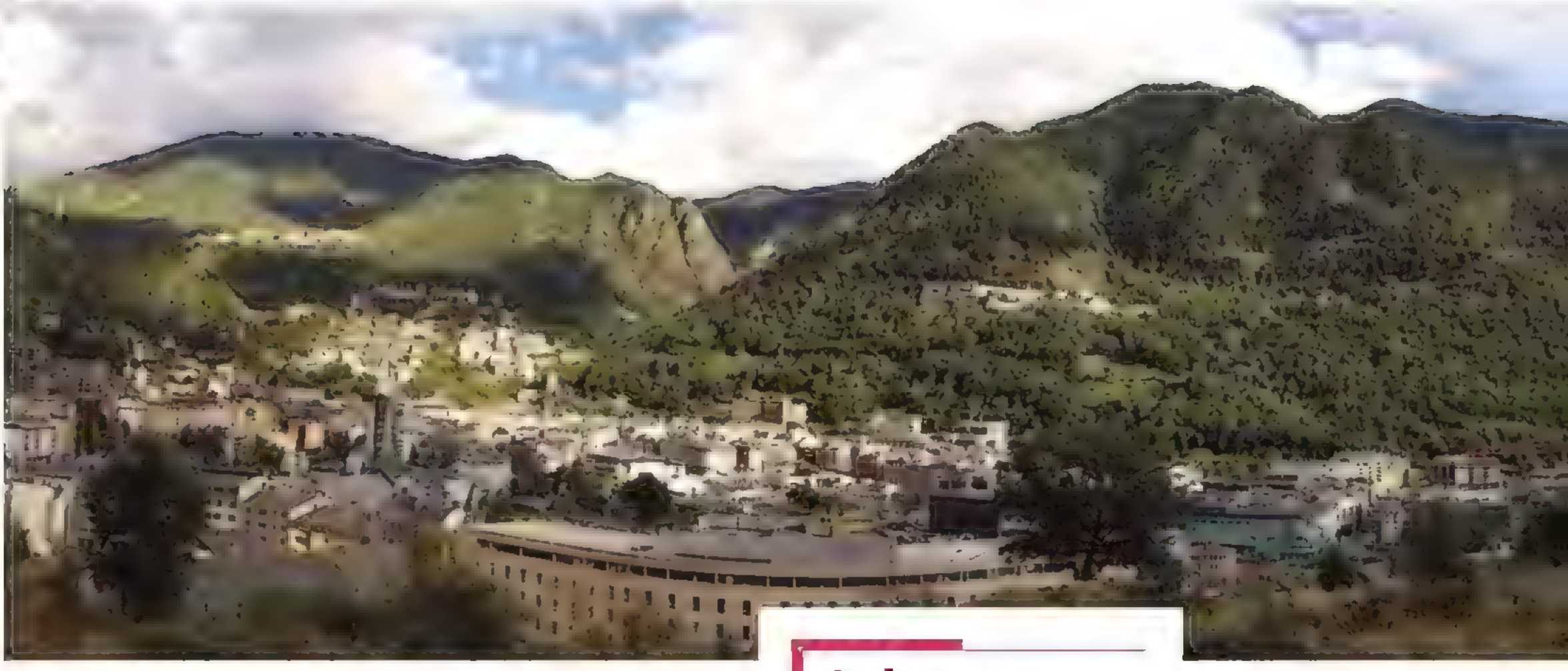
"I usually just describe myself as a 'content creator'," Al tells us. "But to [be specific], I'm not only a music producer, but a DJ, a sound engineer, a streamer, and I have a radio chain, and two stores: one for clothing and one for headbands."

Al was on the stream for only a short time, showing off their robot in the process, so we decided to catch up with them and ask more in the process.

▶ Also a DJ, Al has some radio stations under the name Impulse Radio

What do you like to make?
Essentially anything that has to do with computers, whether that's apps, websites, games, music, designs... So pretty much everything.

> "I'm not only a music producer, but a DJ, a sound engineer, a streamer"



▲ Al lives in Andorra, a small mountainous country located between France and Spain

What do you like to stream?
On my Twitch channel (popito57) I mainly stream two things: music mixes and video games. I don't really do much else there as for me it's a pretty informal platform.

When did you first learn about Raspberry Pi?
I'd say when I lived in the UK and joined the Code Club. I asked where the resources came from and discovered Raspberry Pi.

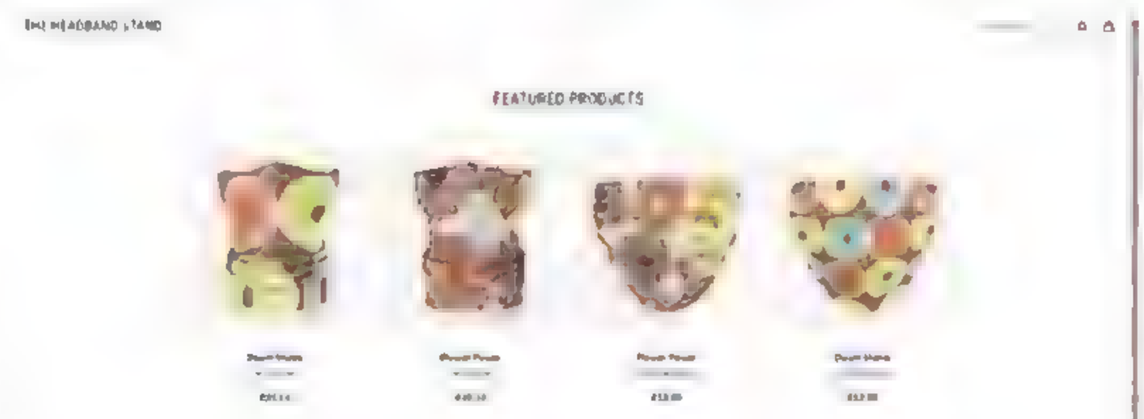
What is the maker community like in Andorra?
I'd mainly say that it focuses more on robots and video games, since that's what the main academy teaches.

What's your favourite thing that you've made?
It's not completely made yet, but I'd say one of my projects, 'The Rainbow Mailbox', which is a messaging hotline for any LGBT kids that are struggling with anything, whether that is in school, with family, etc.

Anything you would like to plug?
I recently made a Linktree with all of my work and social media, including things I didn't/forgot to mention in the stream, such as my Spotify, so check it out if you want to/have the time: **linktr.ee/popito**.

Andorra

A small country of nearly 80,000 people, Andorra lies between Spain and France in the Pyrenees – the average elevation of the country is 1996 metres, and the capital Andorra la Vella is the highest capital city in Europe at 1023 metres! While Catalan is the official language, due to its size and proximity to France and Spain, French and Spanish are also commonly spoken, along with some Portuguese.



▲ We're very partial to the donut masks Al makes here

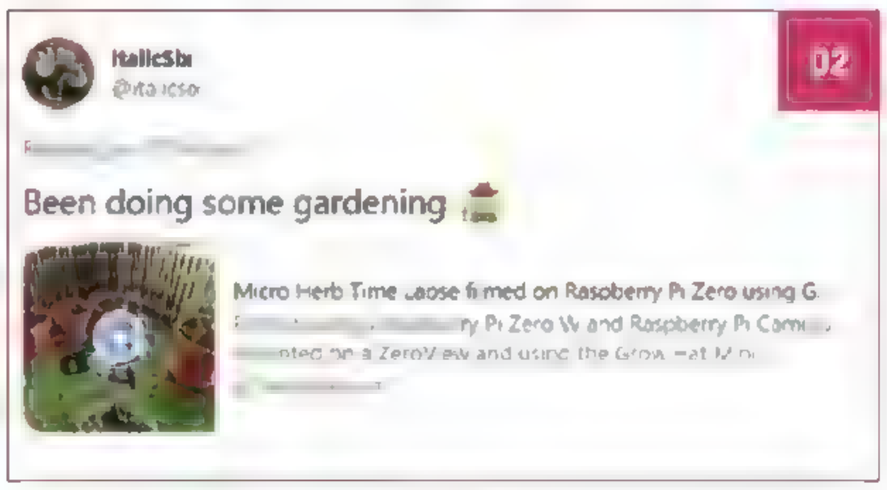
This Month in Raspberry Pi

MagPi Monday

Amazing projects direct from our Twitter!

Every Monday we ask the question: have you made something with a Raspberry Pi over the weekend? Every Monday, our followers send us amazing photos and videos of the things they've made.

Here's a selection of some of the awesome things we got sent this month – and remember to follow along at the hashtag #MagPiMonday!



01. We love the idea of this! And pre-programmed Nyancat is also great
02. A classic and excellent use of time-lapse photography
03. These drones have been kicking up on a storm within the community. They look great
04. PiMowBot continues its progress to lawn cutting domination
05. A smart-looking time-lapse setup from Spencer Organ here
06. Custom controllers are a great way to add something extra to game playing
07. A multi-use project that makes the most of a small space? We love to see it
08. The galloping horse is one of the first major pieces of animation, and it looks great this way
09. Drifting so hard it will enter the D-Dimension
10. This retro-tastic weather robot, aptly named WeatherBot, needs to be a kit we think
11. Curves are tricky. We believe in you, though keep trying

04

TGD
@TGD_Consulting

Enhanced one of the cool summer project from the latest @MagPi issue 106.

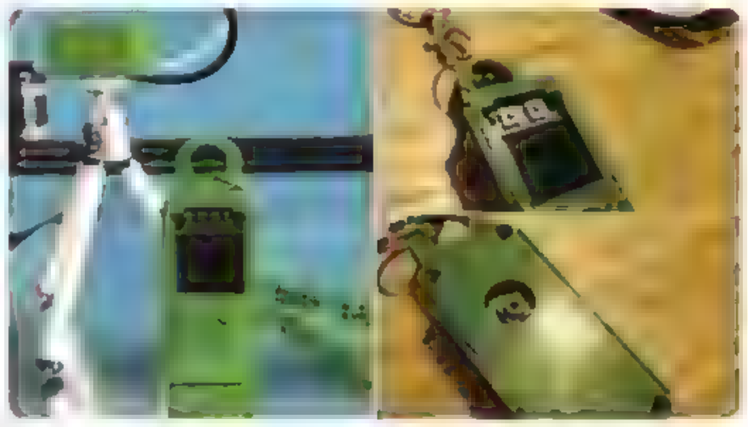
#Make #DIY #3Dprinted #IoT #smart #robotics ... based on #RaspberryPi #MagPiMond... #MagPiTuesday.

Here is a PiMowBot community build in action ... YouTube

05

S Organ #E626FA
@makecupboard

My timelapse camera



06

07

maxi14x

Prepared the balcony with a @flightradar24 antenna, @pimoroni growhat on a @Raspberry Pi zeroW with auto-watering and a @esphome_ temperature sensor.

08

Brian Corteil

Created an Art

'Galloping Horse in 256 Pixels'

Eadweard Muybridge study of a galloping horse animated displayed on a @pimoroni Unicorn HD hat on a @Raspberry Pi P-3 A

09

Dr Feelbug - Roboteer

I made a #3Dprinted drifting tank

...gPi Monday

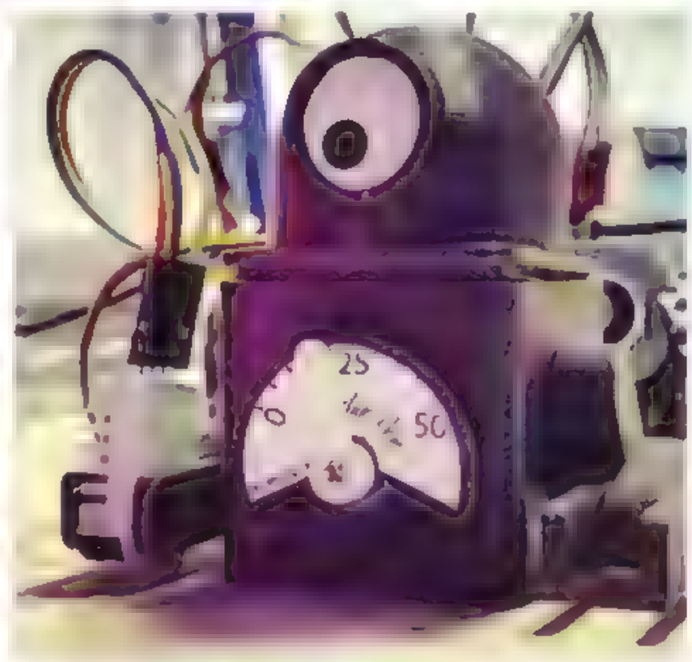
Dr Feelbug - Roboteer ... May 26
...e made a drift tank!

10

Kevin McAleer

Replying to @TheMagPi

I created WeatherBot- a Pico Powered Temperature Robot that displays the room temperature on his belly using a SG90 Servo! I also created a text Adventure Game in MicroPython that you can play via the REPL! (youtu.be/304YCix58Mo)



11

Talleres de Marquesina

Still working on that curve!

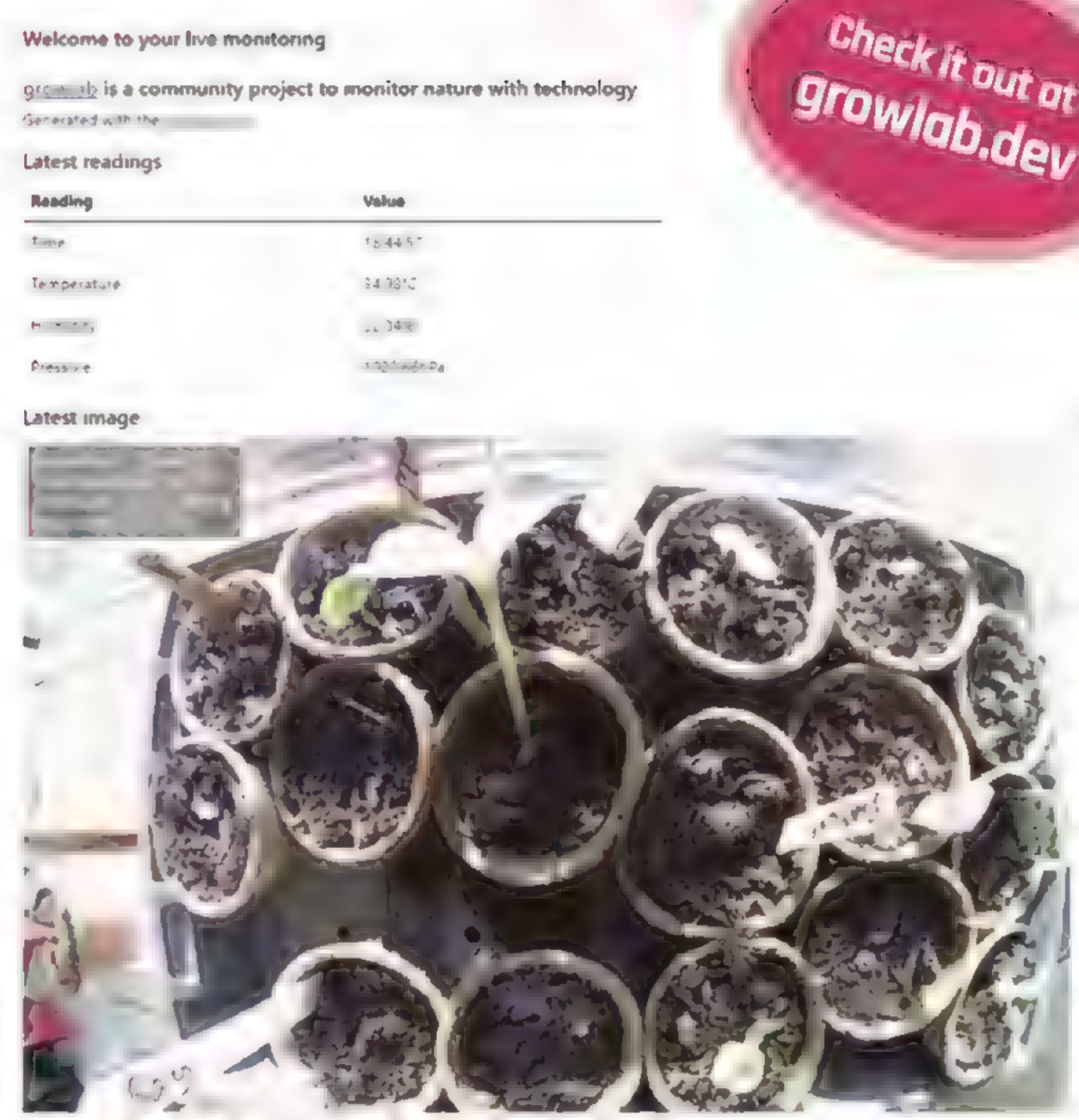
GrowLab

There's a little growing competition going on within the Raspberry Pi community at the moment operating under the hashtag #growlab. The brainchild of Alex Ellis, a mainstay of the community, it's a contest to grow seeds as big as you can before the UK Harvest Festival, which is 23 September this year.

It requires a really simple set of hardware to oversee your plants – namely a Raspberry Pi and a Camera Module. With it, you can track the growth using the camera, and you can easily add extra sensors to try to help automate the growing.

"Everyone is welcome to join #growlab," Alex says on the site. "Whether you're participating in the time-lapse contest, or just wanting to connect with others and grow your own. You'll be able to get started for under 30 dollars, less if you already own a Raspberry Pi and a Camera Module."

You can find out more at **growlab.dev**, which has plenty of info on how to get started with both the tech and the seeds.



▲ You can share your setup over the web for all to see

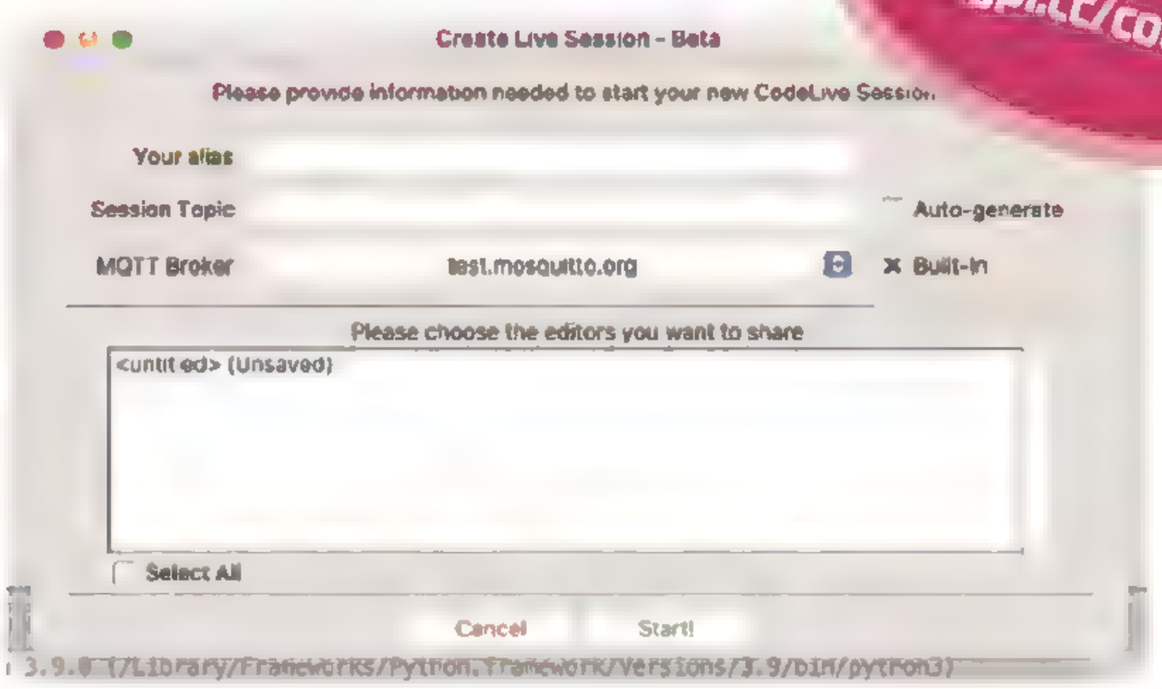
▼ With the right sensors you can track a lot of minutiae, perfect for improving your growing

Thonny Online Collab

Find out more at magpi.cc/codelive

One of the emails we got this month was from a group of students graduating from Calvin University in Michigan, USA. Brad Ritzema, Samuel Zeleke, and Enoch Mwesigwa created a live collaboration plug-in for the Thonny Python IDE. It basically allows you to code with other people.

Their paper addresses the issues with teaching industry practices using online learning. "We sought to rectify this problem by introducing a live collaboration plug-in titled CodeLive for Thonny. [It supports] pair programming where one user (the 'driver') writes code while an indeterminate number of other users (the 'navigators') view their changes. This takes place while the users converse over an external voice/video channel of their choice."



Your Letters

Backwards compatibility

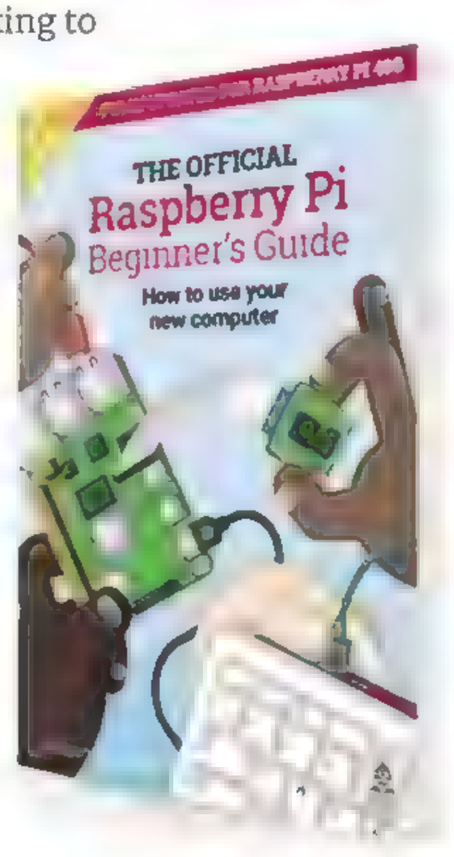
I bought a Raspberry Pi 1 Model B some years ago from Maplins. I would like to buy a book for beginners. Will any of the beginner guides still be suitable for this older model?

Colin via email

So largely, the code and programs are backwards compatible with even that first Raspberry Pi Model B. Some stuff may run a little slower, and without the full 40 pin GPIO you won't be able to attach all available add-ons.

However, if you're still looking to learn to code, work with electronics, or maybe even set up a media centre, you definitely can.

We recommend our Beginner's Guide which you can find here: **magpi.cc/bguide4**.



RP2040 for all

I noticed that RP2040 chips are now being sold on their own, so I grabbed some for my own experiments. Have other companies got these lovely chips and managed to make anything with them yet? I'd gladly pay for a souped-up Raspberry Pi Pico.

Kirsty via Facebook

There's been a fair few products released using RP2040, from boards that are smaller than a Raspberry Pi Pico like the Tiny 2040 from Pimoroni (**magpi.cc/tiny2040**), Arduino Nano 2040 which has wireless LAN (**magpi.cc/nano2040**), and others from Adafruit and SparkFun.

There's plenty of other types of products, like the Keybow 2040 from Pimoroni, which you can use as a macro keyboard. Check them out!

Not so big builds

I'm not sure quite what I expected but as it turns out, the projects in the Big Builds article were a little too big for me. Is there any chance that you'd do a slightly larger builds article? Or even a normal builds article?

Jon via Twitter

You're in luck, this issue we have a follow-up to Big Builds called Handheld Builds – all part of Feature's Editor Rob's Big Build label. You can find them starting on page 72, for some great ideas on making small things that take a bit of work.

Contact us!

- Twitter **@TheMagPi**
- Facebook **magpi.cc/facebook**
- Email **magpi@raspberrypi.com**
- Online **raspberrypi.org/forums**

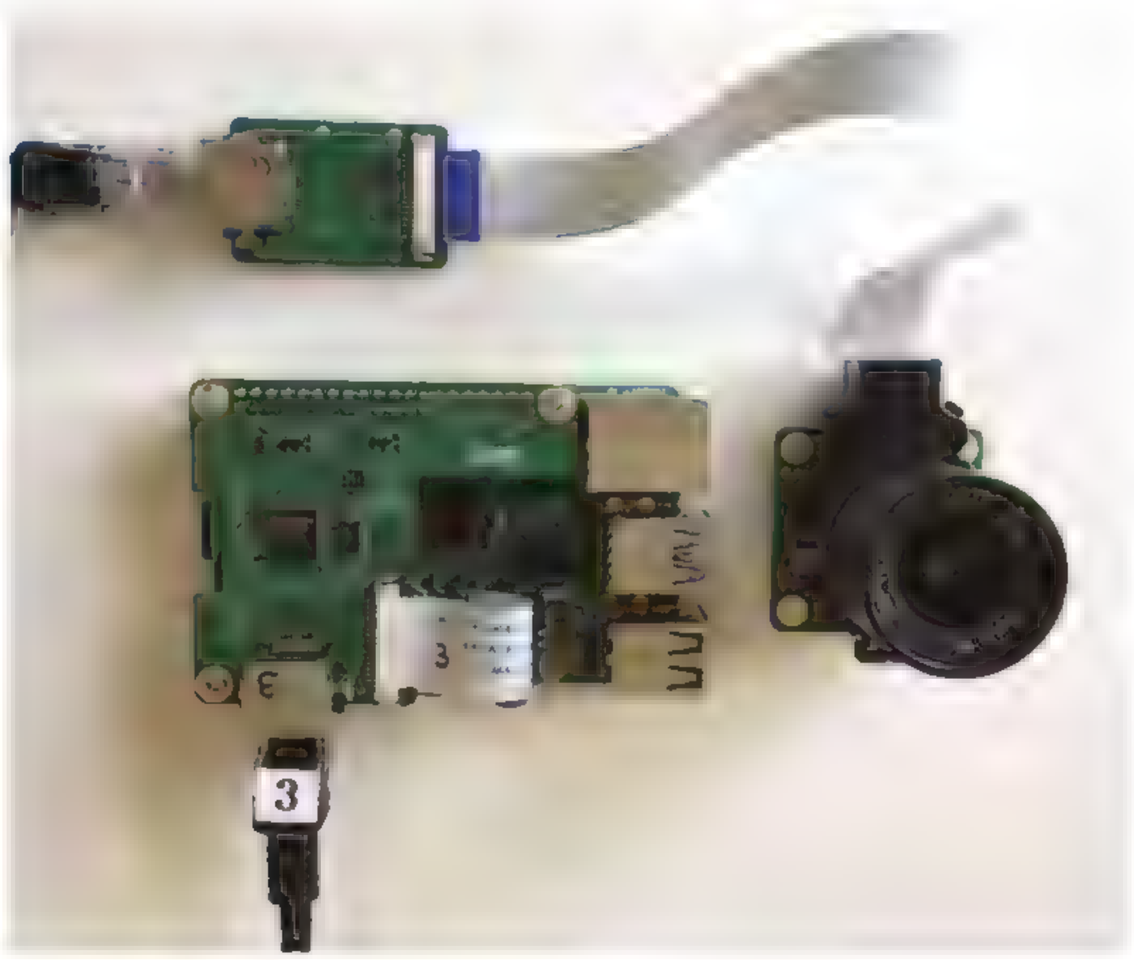
Wireframe

Join us as we lift the lid on video games

Visit **wfmag.cc** to learn more

WIN ONE OF FIVE THINE CAMERA EXTENSION KITS IN ASSOCIATION WITH THine

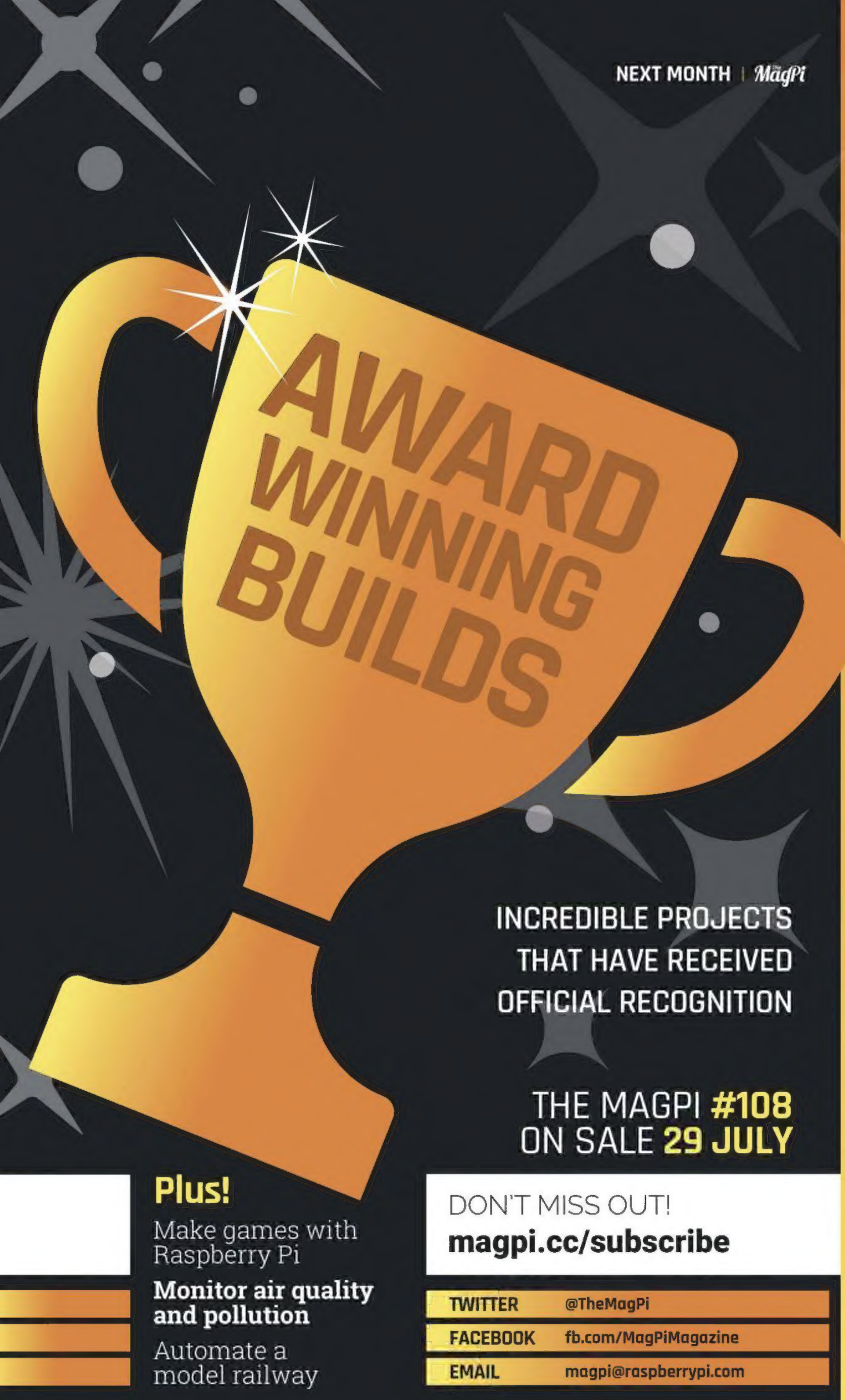
This LAN cable extension kit allows you to connect a Raspberry Pi Camera Module or HQ Camera from up to 20 m away without any quality loss! We reviewed it last issue and gave it 9/10, and now you can win one.



Head here to enter: magpi.cc/win | Learn more: magpi.cc/cableextender

Terms & Conditions

Competition opens on **23 June 2021** and closes on **29 July 2021**. Prize is offered to participants worldwide aged 13 or over, except employees of the Raspberry Pi Foundation, Raspberry Pi Trading (Ltd), the prize supplier, their families, or friends. Winners will be notified by email no more than 30 days after the competition closes. By entering the competition, the winner consents to any publicity generated from the competition, in print and online. Participants agree to receive occasional newsletters from The MagPi magazine. We don't like spam: participants' details will remain strictly confidential and won't be shared with third parties. Prizes are non-negotiable and no cash alternative will be offered. Winners will be contacted by email to arrange delivery. Any winners who have not responded 60 days after the initial email is sent will have their prize revoked. This promotion is in no way sponsored, endorsed or administered by, or associated with, Instagram or Facebook.



INCREDIBLE PROJECTS
THAT HAVE RECEIVED
OFFICIAL RECOGNITION

THE MAGPI **#108**
ON SALE **29 JULY**

Plus!

Make games with Raspberry Pi

Monitor air quality and pollution

Automate a model railway

DON'T MISS OUT!
magpi.cc/subscribe

TWITTER	@TheMagPi
FACEBOOK	fb.com/MagPiMagazine
EMAIL	magpi@raspberrypi.com

EDITORIAL

Editor
Lucy Hattersley
lucy@raspberrypi.com

Features Editor
Rob Zwetsloot
rob@raspberrypi.com

Sub Editor
Nicola King

ADVERTISING

Charlotte Milligan
charlotte.milligan@raspberrypi.com
+44 (0)7725 368887

DESIGN

criticalmedia.co.uk

Head of Design
Lee Allen

Designers
Lucy Cowan, Sam Ribbits

Illustrator
Sam Alder

CONTRIBUTORS

Wes Archer, Mike Cook, David Crookes, Gareth Halfacree, Martin O'Hanlon, Rosemary Hattersley, Nicola King, KG Orphanides, Nik Rawlinson, Laura Sach, Mark Vanstone

PUBLISHING

Publishing Director
Russell Barnes
russell@raspberrypi.com

Director of Communications
Liz Upton

CEO
Eben Upton

DISTRIBUTION

Seymour Distribution Ltd
2 East Poultry Ave,
London EC1A 9PT
+44 (0)207 429 4000

SUBSCRIPTIONS

Unit 6 The Enterprise Centre
Kelvin Lane, Manor Royal,
Crawley, West Sussex, RH10 9PE
+44 (0)1293 312193
magpi.cc/subscribe

recycle

This magazine is printed on paper sourced from sustainable forests and the printer operates an environmental management system which has been assessed as conforming to ISO 14001.

The MagPi magazine is published by Raspberry Pi (Trading) Ltd., Maurice Wilkes Building, St. John's Innovation Park, Cowley Road, Cambridge, CB4 0DS. The publisher, editor, and contributors accept no responsibility in respect of any omissions or errors relating to goods, products, or services referred to or advertised in the magazine. Except where otherwise noted, content in this magazine is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported (CC BY-NC-SA 3.0).

Raspberry Pi PRESS

ISSN: 2051-9982.

Compatibility

Old Raspberry Pi computers still have a lot of life left in them, according to **Rob Zwetsloot**

We had an email this month from a reader who had an original Raspberry Pi Model B, and was wondering if he could still use it. You can read more about it on our letters pages (92–93), but the short answer is: yes!

It got me thinking about how I use/have used older Raspberry Pi in the past. Up until recently, I was using a Raspberry Pi 2 as my in-home file server with no problems. I still have a NES controller that I use for NES emulation, that has a pre-production Raspberry Pi Zero inside it. I have a Raspberry Pi 3 that powers my interactive tabletop RPG screen, which hopefully I'll be able to use again soon.

Demystify it

Due to my position here at *The MagPi*, I do have a wider variety of Raspberry Pi boards than most; however, you can always grab cheaper/older models off eBay, or your auction/local sales website of choice, if you're on a really tight budget.

This extended lifespan is by design. Of course, code is code, so learning and using Python naturally works across different Raspberry Pi boards. However, even the newest Raspberry Pi OS is still compatible with original Raspberry Pi models. In the past, when I've spoken with engineers in the office, they've pointed out how a

> Up until recently, I was using a Raspberry Pi 2 as my in-home file server with no problems

lot of features and functions are backwards compatible. Some still have an original Raspberry Pi Zero powering Kodi on a TV at home.

It's clear that power isn't everything. Just look at Raspberry Pi Pico – it's more suited to certain projects than even the most powerful Raspberry Pi, and it's much smaller to boot.

Recycling Raspberry Pi

Using and reusing Raspberry Pi is built into its DNA. The amount of different little projects I've put together with the same Raspberry Pi is too high to count (on our fingers at least), and just because there's a newer Raspberry Pi, it doesn't mean we've stopped using it. I think I've only broken a single Raspberry Pi, and that was an original Model B that got a lot of use in the early years from a lot of people. The only reason I upgraded my file server Raspberry Pi is because the Raspberry Pi 2 became a gift for my sister, and she still uses it to this day.

Upcycling doesn't just have to mean using hardware from the eighties, and recycling doesn't have to mean completely melting down and remaking – Raspberry Pi computers have a very long lifespan, and can be recycled ad infinitum for newer and better projects thanks to cross-generational compatibility.

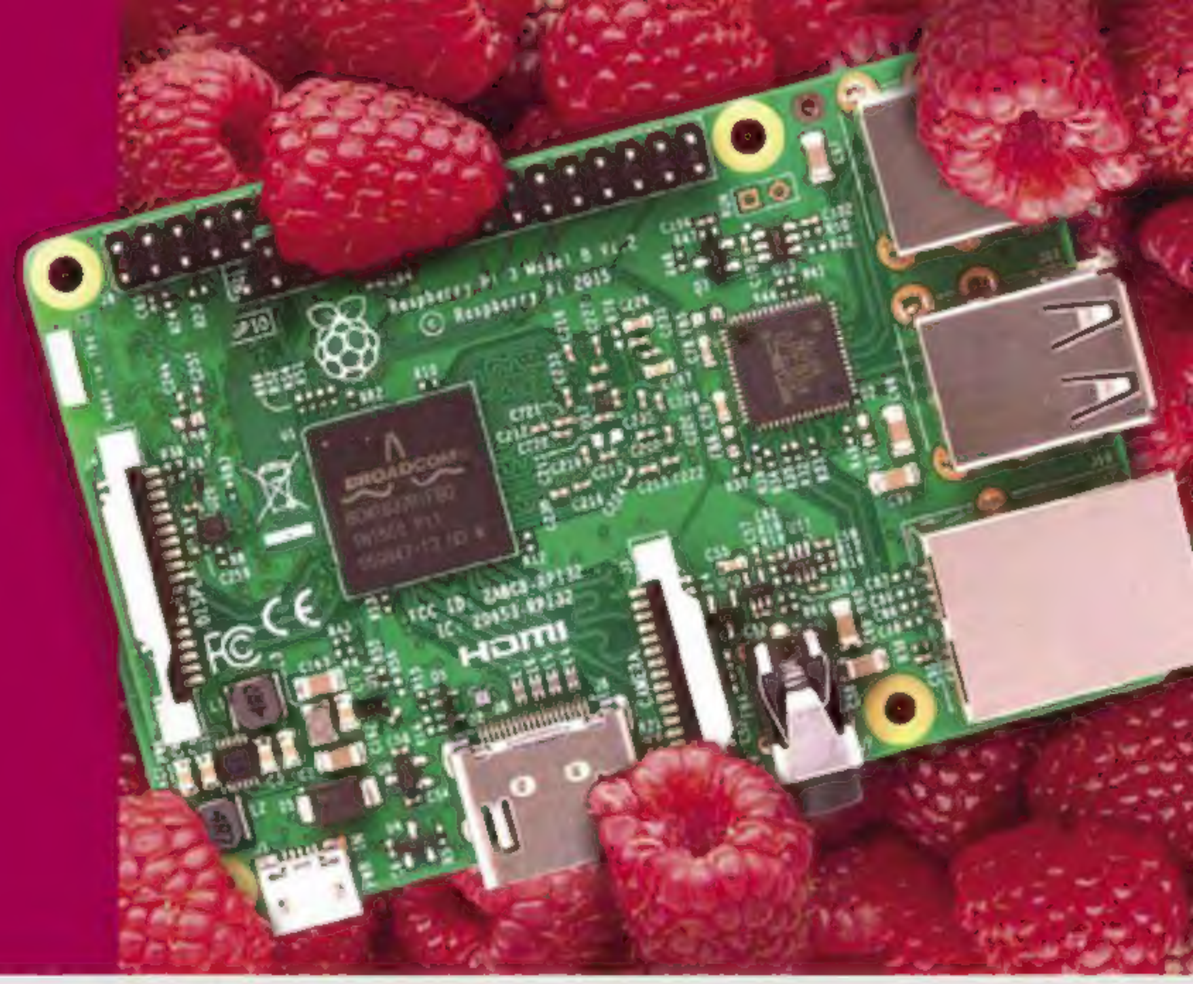
AUTHOR

Rob Zwetsloot

Rob is amazing. He's also the Features Editor of *The MagPi*, a hobbyist maker, cosplayer, comic book writer, and extremely modest.

magpi.cc

American **Raspberry Pi** Shop



- Displays
- HATs
- Sensors
- Cases
- Arcade
- Swag
- Project Kits
- Cameras
- Power Options
- Add-on Boards
- Cables and Connectors
- GPIO and Prototyping

Partner and official reseller for top Pi brands:

adafruit · sparkfun. ELECTRONICS · PIMORONI · HiFiBerry · and many others!

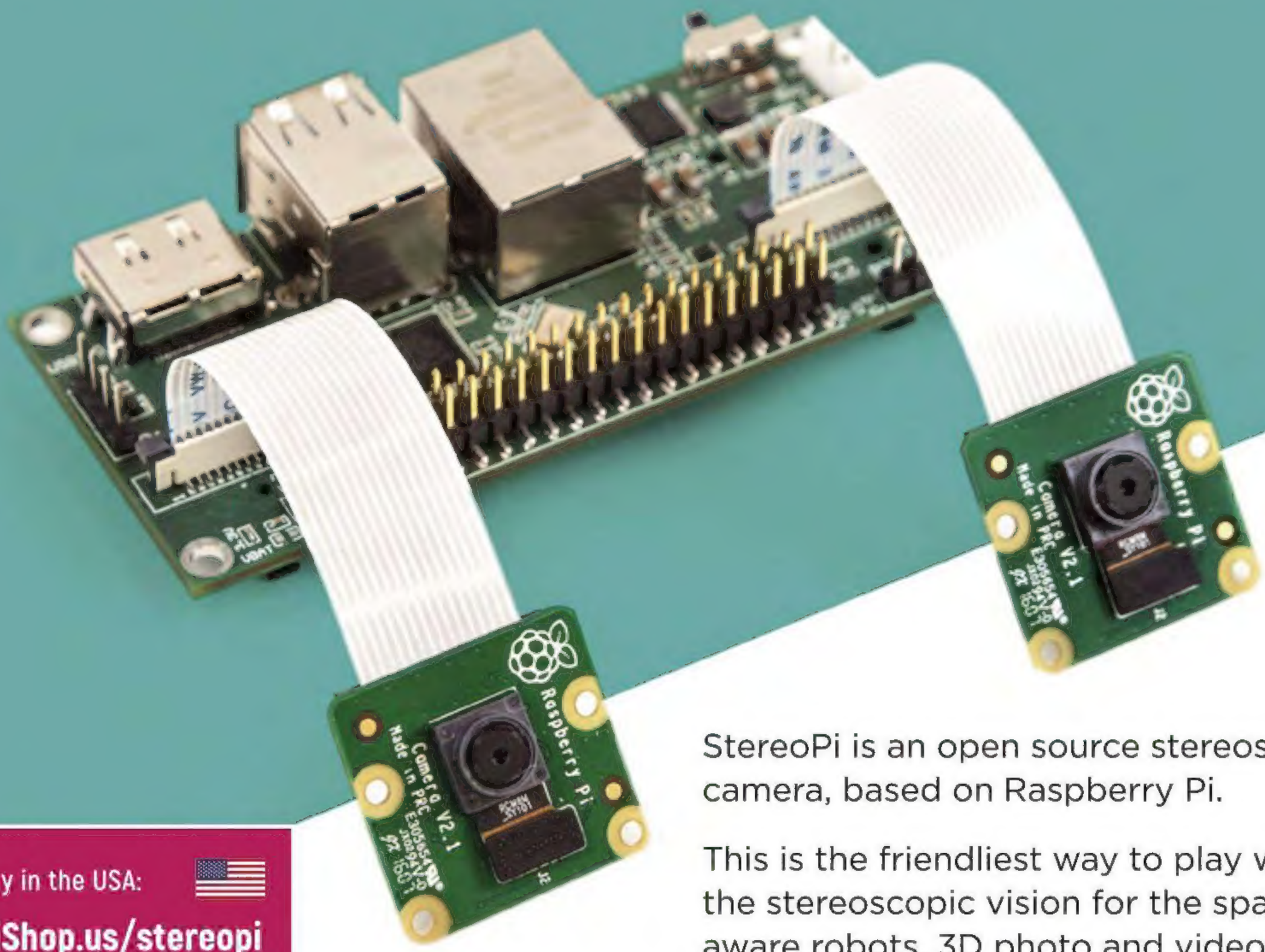
Price, service, design, and logistics support for **VOLUME PROJECTS**

USA **PiShop.us**

Canada **BuyaPi.ca**

Raspberry Pi APPROVED RESELLER

Do you know
HOW ROBOTS SEE?



StereoPi is an open source stereoscopic camera, based on Raspberry Pi.

This is the friendliest way to play with the stereoscopic vision for the spatially aware robots, 3D photo and video!

Buy in the USA:

PiShop.us/stereopi

Buy in Canada:

BuyaPi.ca/stereopi

RASPBERRY PI INSIDE

Raspbian

STOCK RASPBIAN SUPPORT

OPEN SOURCE

CROWDFUNDED PROJECT

LinuxGizmos.com
"The StereoPi can capture, save, livestream, and process real-time stereoscopic video and images for robotics, AR/VR, computer vision, drone instrumentation, and panoramic video."

MickMake
"With it you can do things like, stream stereoscopic 3D video to YouTube, build real-time depth maps using OpenCV, create panoramics using Hugin and even a 3rd person view of real life. Cool."

Raspberry Pi Blog
"There are some excellent community efforts too, of which our current favourite is this nifty dual camera board."

Hackster News
"You can hook this up to YouTube, to Oculus Go, you can use it with OpenCV.. I cannot wait to start messing around with these because it's basically a dream come true."

Elettronica In · LinuxGizmos.com · Geeky Gadgets · OPENELECTRONICS · electronics-lab